AFRL-IF-RS-TR-1999-225, Volume II (of two)
Final Technical Report
October 1999

KBSA LIFE CYCLE EVALUATION

USC Center for Software Engineering

Barry Boehm, A. Winsor Brown and Prasanta Bose

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-1999-225, Volume II (of two) has been reviewed and is approved for publication.

Douglas A. White

APPROVED:
DOUGLAS A. WHITE
Project Engineer

FOR THE DIRECTOR:
NORTHRUP FOWLER, III, Technical Advisor
Information Technology Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFTD, 525 Brooks Rd, Rome, NY 13441-4505. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
Oct 99

3. REPORT TYPE AND DATES COVERED
Final Sep 96 - Aug 98

4. TITLE AND SUBTITLE
KBSA LIFE CYCLE EVALUATION

5. FUNDING NUMBERS
C - F30602-96-C-0274
PE - 62702F
PR - 5581
TA - 27
WU - 96

6. AUTHOR(S)
Barry Boehm, A. Winsor Brown, and Prasanta Bose

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
USC Center for Software Engineering
Computer Sciences Department
University of Southern California
Los Angeles, CA 90089-0781

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFRL/IFTD
525 Brooks Rd
Rome, NY 13441-4505

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-1999-225
Vol II (of two)

11. SUPPLEMENTARY NOTES
AFRL Project Engineer: Douglas White, IFTD, 315-330-2129

12a. DISTRIBUTION AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The objective of this research effort was to develop and validate technical approaches for evaluating the effects of Knowledge-Based Software Assistant (KBSA) and Evolutionary Design of Complex Software (EDCS) process concepts and technology on software development effort and schedule, and to use these technical approaches to perform comparative evaluations of technology. The approach taken for this research included four tasks. The first was to characterize sources of software technology in the context of recent and emerging software trends. The second was to develop models and an evaluation framework providing a baseline for assessing the effects of software technology on software development effort and schedule. The third task used these models to evaluate KBSA, EDCS and commercial technology with respect to the baseline. The fourth task formulated conclusions and recommendations based upon the results of the study. The report is divided into two volumes because of its length. The first volume documents the study and summarizes the data and results. The second volume provides additional detail on the models and the evaluations performed.

14. SUBJECT TERMS
metrics, models, life cycle evaluation, software, knowledge-based design, evolutionary design

15. NUMBER OF PAGES
162

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
UNCLASSIFIED

18. SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

19. SECURITY CLASSIFICATION OF ABSTRACT
UNCLASSIFIED

20. LIMITATION OF ABSTRACT
UL

J.1 Table of Contents
Volume II

Table of contents (continued)

Table of contents (continued)

J.2 Table of Figures

K. Appendix 1

L. Appendix 2, Part 1

N. Appendix 4

O. Appendix 5

J.3 List of Tables

K. Appendix 1

L. Appendix 2, Part 1

K. Appendix 1.

COCOMO Summary

USC-CSE COCOMO Team

1 Introduction

COCOMO II is a parametric model for software cost/estimation.

The two main elements of the COCOMO II strategy are:

- Preserve the openness of the original COCOMO with all of its relationships and algorithms publicly available.
- Key the inputs and outputs of the COCOMO II submodels to the level of information available.

All of its interfaces are designed to be public, well-defined, and parametrized so that complementary preprocessors (analogy, case-based, or other size estimation models), post-processors (project planning and control tools, project dynamics models, risk analyzers), and higher level packages (project management packages, product negotiation aids) can be combined straightforwardly with COCOMO II.

2 Overall Model Definition

2.1 COCOMO II Models for the Software Marketplace Sectors

The COCOMO II capability for estimation of Application Generator, System Integration, or Infrastructure developments is based on two increasingly detailed estimation models for subsequent portions of the life cycle, *Early Design* and *Post-Architecture*.

2.1.1 COCOMO II Model Rationale and Elaboration

This mix of models rests on three primary premises: First, current and future software projects will be tailoring their processes to their particular process drivers. These process drivers include: COTS or reusable software availability; degree of understanding of architectures and requirements; market window or other schedule constraints; size; and required reliability (see [Boehm 1989, pp. 436-37] for an example of such tailoring guidelines). Second, the granularity of the software cost estimation model used needs to be consistent with the granularity of the information available to support software cost estimation. Third, COCOMO II does not produce point estimates of software cost and effort, but rather ranges estimates tied to the granularity of the estimation inputs.

With respect to process strategy, Application Generator, System Integration, and Infrastructure software projects will involve a mix of three major process models. The appropriate models will depend on the project marketplace drivers and degree of product understanding.

The Early Design model involves exploration of alternative software/system architectures and concepts of operation. At this stage, not enough is generally known to support fine-grain cost estimation. The corresponding COCOMO II capability involves the use of function points and a course-grained set of 7 cost drivers. (e.g. Two cost drivers for Personnel Capability and Personnel Experience in place of the 6 COCOMO II Post-Architecture model cost drivers covers various aspects of personnel capability, continuity, and experience.)

The Post-Architecture model involves the actual development and maintenance of a software product. This stage proceeds most cost-effectively if a software life-cycle architecture has been developed; validated with respect to the system's mission, concept of operation, and risk; and established as the framework for the product. The corresponding COCOMO II model has about the same granularity as the previous COCOMO and Ada COCOMO models. It uses source instructions and/or function points for sizing with modifiers for reuse and software breakage, a set of 17 multiplicative cost drivers, and a set of 5 factors determining the project's scaling exponent. These factors replace the development modes (Organic, Semidetached, or Embedded) in the original COCOMO model, and refine the four exponent-scaling factors in Ada COCOMO.

To summarize, COCOMO II provides the following three-stage series of models for estimation of Application Generator, System Integration, and Infrastructure software projects:

1. The earliest phases or spiral cycles will generally involve prototyping, using the Application Composition model capabilities. The COCOMO II Application Composition model supports these phases, and any other prototyping activities occurring later in the life cycle.

2. The next phases or spiral cycles will generally involve exploration of architectural alternatives or incremental development strategies. To support these activities, COCOMO II provides an early estimation model called the Early Design model. This level of detail in the model is consistent with the general level of information available and the general level of estimation accuracy needed at this stage.

3. Once the project is ready to develop and sustain a fielded system, it should have a life-cycle architecture, which provides more accurate information on cost driver inputs, and enables more accurate cost estimates. To support this stage, COCOMO II provides the Post-Architecture model.

2.2 Development Effort Estimates

In COCOMO II effort is expressed as Person Months (PM). All effort equations are presented in "COCOMO II Model Definition Manual." A person month is the amount of time one person spends working on the software development project for one month. This number is exclusive of holidays and vacations but accounts for weekend time off. The number of person months is different from the time it will take the project to complete; this is called the development schedule. For example, a project may be estimated to require 50 PM of effort but have a schedule of 11 months.

Equation 2.1 is the base model for the Early Design and Post-Architecture cost estimation models. The inputs are the *Size* of software development, a "constant," *A*, and a scale factor, *B*. The size is in units of thousands of source lines of code (KSLOC). This is derived from estimating the size of software modules that will constitute the application program. It can also be estimated from unadjusted function points (UFP), converted to SLOC then divided by one thousand. Procedures for counting SLOC or UFP are explained in the chapters on the Post- Architecture and Early Design models respectively.

The scale (or exponential) factor, *B*, accounts for the relative economies or diseconomies of scale encountered for software projects of different sizes [Banker et al 1994a]. The "constant", *A*, is used to capture the multiplicative effects on effort with projects of increasing size

$$PM_{nominal} = A \times (Size)^{B} \qquad (2.1)$$

2.3 Software Economies and Diseconomies of Scale

Software cost estimation models often have an exponential factor to account for the relative economies or diseconomies of scale encountered in different size software projects. The exponent, *B*, in Equation 2.1 is used to capture these effects.

If $B < 1.0$, the project exhibits economies of scale. If the product's size is doubled, the project effort is less than doubled. The project's productivity increases as the product size is increased. Some project economies of scale can be achieved via project-specific tools (e.g., simulations, testbeds) but in general these are difficult to achieve. For small projects, fixed start-up costs such as tool tailoring and setup of standards and administrative reports are often a source of economies of scale.

If $B = 1.0$, the economies and diseconomies of scale are in balance. This linear model is often used for cost estimation of small projects. It is used for the COCOMO II *Applications Composition* model.

If $B > 1.0$, the project exhibits diseconomies of scale. This is generally due to two main factors: growth of interpersonal communications overhead and growth of large-system integration overhead. Larger projects will have more personnel, and thus more interpersonal communications paths consuming overhead. Integrating a small product as part of a larger product requires not only the effort to develop the small product, but also the additional overhead effort to design, maintain, integrate, and test its interfaces with the remainder of the product.

2.3.1 Basis of Economy and Diseconomy of Scale

Historically, project and target platform environment impact the diseconomies of scale in software development. Embedded-software projects tended to be more unprecedented, requiring more communication overhead and complex integration, and less flexible, requiring more communications overhead and extra effort to resolve issues within tight schedule, budget, interface, and performance constraints. Conversely communications overhead and integration overhead can be reduced significantly by early risk and error elimination; by using thorough, validated architectural specifications; and by stabilizing requirements.

As a result COCOMO II's model added into the scale factors for the architecture and risk factors as a single factor. The COCOMO II model also has a process maturity factor based on the Software Engineering Institute (SEI) definition. The model also includes precedentedness and flexibility factors and a Team Cohesiveness factor to account for the diseconomy-of-scale effects on software projects whose developers, customers, and users have difficulty in synchronizing their efforts.

2.3.2 Scaling Drivers

Equation 2.2 defines the exponent, *B*, used in Equation 2.1. Table 2-1, which follows the discussion of the arithmetic, shows the rating levels for the COCOMO II scale drivers. The selection of scale drivers is based on the rationale that they are a significant source of exponential variation on a project's effort or productivity variation. Each scale driver has a range of rating levels, from Very Low to Extra High. Each rating level has a weight, *W*, and the specific value of the weight is called a scale factor. A project's scale factors, W_i, are summed across all of the factors, and used to determine a scale exponent, *B,* via the following formula:

$$B = .91 + 0.01 \times \sum W_i \tag{2.2}$$

[Data for following examples in "CII98-Official+Anal.xls"]For example, if scale factors with an Extra High rating are each assigned a weight of (0), then a 100 KSLOC project with Extra High ratings for all factors will have $\Sigma W_i = 0$, $B = 0.91$, and a relative effort $E = 100^{.91} = 66\ PM$. If scale factors with Very Low rating are each assigned a weight of (5), then a project with Very Low (5) ratings for all factors will have $\Sigma W_i = 31.62$, $B = 1.226$ and a relative effort $E = 283.4\ PM$. This represents a large variation, but the increase involved in a one-unit change in one of the factors is only about 4.7%.

Table 2-1: Scale Factors for COCOMO II Early Design and Post-Architecture Models

Scale Factors (W_i)	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprecedented	largely unprecedented	somewhat unprecedented	Generally familiar	largely familiar	thoroughly familiar
FLEX	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
RESL[1]	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
PMAT	Weighted average of "Yes" answers to CMM Maturity Questionnaire					

2.3.3 Values for Scale Factors

Section 4.0 (page 29) contains the numeric values for the COCOMO II-1998 Scale Factors. The same values are used for the Early Design and the Post Architecture models. The rest of this section discusses the scale factors in terms of their nominal levels.

2.3.3.1 Precedentedness (PREC) and Development Flexibility (FLEX)

Table 2.2 maps project features onto the Precedentedness and Development Flexibility scales. This table can be used as a more in depth explanation for the PREC and FLEX rating scales given in Table 2-1.

[1] % significant module interfaces specified,% significant risks eliminated.

Table 2-2: Scale Factors Related to PREL and FLEX

Feature	Very Low	Nominal / High	Extra High
Precedentedness (PREC)			
Organizational understanding of product objectives	General	Considerable	Thorough
Experience in working with related software systems	Moderate	Considerable	Extensive
Concurrent development of associated new hardware and operational procedures	Extensive	Moderate	Some
Need for innovative data processing architectures, algorithms	Considerable	Some	Minimal
Development Flexibility (FLEX)			
Need for software conformance with pre-established requirements	Full	Considerable	Basic
Need for software conformance with external interface specifications	Full	Considerable	Basic
Premium on early completion	High	Medium	Low

2.3.3.2 Architecture / Risk Resolution (RESL)

This factor combines two scale factor concepts, “Design Thoroughness by Product Design Review (PDR)” and “Risk Elimination by PDR” [Boehm and Royce 1989; Figures 4 and 5]. Table 2-3 consolidates the concepts to form a comprehensive definition for the RESL rating levels. The RESL rating is the subjective weighted average of the listed characteristics.

Table 2-3: RESL Rating Components

Characteristic	Very Low	Low	Nominal	High	Very High	Extra High
Risk Management Plan identifies all critical risk items, establishes milestones for resolving them by PDR.	None	Little	Some	Generally	Mostly	Fully
Schedule, budget, and internal milestones through PDR compatible with Risk Management Plan	None	Little	Some	Generally	Mostly	Fully
Percent of development schedule devoted to establishing architecture, given general product objectives	5	10	17	25	33	40
Percent of required top software architects available to project	20	40	60	80	100	120
Tool support available for resolving risk items, developing and verifying architectural specs	None	Little	Some	Good	Strong	Full
Level of uncertainty in Key architecture drivers: mission, user interface, COTS, hardware, technology, performance.	Extreme	Signifi-cant	Consider able	Some	Little	Very Little
Number and criticality of risk items	> 10 Critical	5-10 Critical	2-4 Critical	1 Critical	> 5 Non-Critical	< 5 Non-Critical

2.3.3.3 Team Cohesion (TEAM)

The Team Cohesion scale factor accounts for the sources of project turbulence and entropy due to difficulties in synchronizing the project's stakeholders: users, customers, developers, maintainers, interfacers, others. These difficulties may arise from differences in stakeholder objectives and cultures; difficulties in reconciling objectives; and stakeholder's lack of experience and familiarity in operating as a team. Table 2-4 provides a detailed definition for the overall TEAM rating levels. The final rating is the subjective weighted average of the listed characteristics.

Table 2-4: TEAM Rating Components

Characteristic	Very Low	Low	Nominal	High	Very High	Extra High
Consistency of stakeholder objectives and cultures	Little	Some	Basic	Considerable	Strong	Full
Ability, willingness of stakeholders to accommodate other stakeholders' objectives	Little	Some	Basic	Considerable	Strong	Full
Experience of stakeholders in operating as a team	None	Little	Little	Basic	Considerable	Extensive
Stakeholder teambuilding to achieve shared vision and commitments	None	Little	Little	Basic	Considerable	Extensive

2.3.3.4 Process Maturity (PMAT)

The procedure for determining PMAT is organized around the Software Engineering Institute's Capability Maturity Model (CMM). The time period for rating Process Maturity is the time the project starts. There are two ways of rating Process Maturity. The first captures the result of an organization evaluation based on the CMM (an Assessment or a Capability Evaluation).

Overall Maturity Level

CMM Level 1 (lower half)

CMM Level 1 (upper half)

CMM Level 2

CMM Level 3

CMM Level 4

CMM Level 5

Key Process Areas

The second is organized around the 18 Key Process Areas (KPAs) in the SEI Capability Maturity Model [Paulk et al. 1993, 1993a]. The procedure for determining PMAT is to decide the percentage of compliance for each of the KPAs. If the project has undergone a recent CMM Assessment then the percentage compliance for the overall KPA (based on KPA Key Practice compliance assessment data) is used. If an assessment has not been done then the levels of compliance to the KPA's goals are used (with the Likert scale below) to set the level of compliance. The goal-based level of compliance is determined by a judgement-based averaging across the goals for each Key Process Area.

Table 2-5

Key Process Areas	Almost Always (>90%)	Often (60-90%)	About Half (40-60%)	Occasionally (10-40%)	Rarely If Ever (<10%)	Does Not Apply	Don't Know
Requirements Management	□	□	□	□	□	□	□
Software Project Planning	□	□	□	□	□	□	□
Software Project Tracking and Oversight	□	□	□	□	□	□	□
Software Subcontract Management	□	□	□	□	□	□	□
Software Quality Assurance	□	□	□	□	□	□	□
Software Configuration Management	□	□	□	□	□	□	□
Organization Process Focus	□	□	□	□	□	□	□
Organization Process Definition	□	□	□	□	□	□	□
Training Program	□	□	□	□	□	□	□
Integrated Software Management	□	□	□	□	□	□	□
Software Product Engineering	□	□	□	□	□	□	□
Intergroup Coordination	□	□	□	□	□	□	□
Peer Reviews	□	□	□	□	□	□	□
Quantitative Process Management	□	□	□	□	□	□	□
Software Quality Management	□	□	□	□	□	□	□
Defect Prevention	□	□	□	□	□	□	□
Technology Change Management	□	□	□	□	□	□	□
Process Change Management	□	□	□	□	□	□	□

- Check Almost Always when the goals are consistently achieved and are well-established in standard operating procedures (over 90% of the time).
- Check Frequently when the goals are achieved relatively often, but sometimes are omitted under difficult circumstances (about 60 to 90% of the time).
- Check About Half when the goals are achieved about half of the time (about 40 to 60% of the time).
- Check Occasionally when the goals are sometimes achieved, but less often (about 10 to 40% of the time).
- Check Rarely If Ever when the goals are rarely if ever achieved (less than 10% of the time).
- Check Does Not Apply when you have the required knowledge about your project or organization and the KPA, but you feel the KPA does not apply to your circumstances.
- Check Don't Know when you are uncertain about how to respond for the KPA.

After the level of KPA compliance is determined each compliance level is weighted and a PMAT factor is calculated, as in Equation 2.3. Initially, all KPAs will be equally weighted.

$$5-\left[\sum_{i=1}^{18}\left(\frac{KPA\%_i}{100}\times\frac{5}{18}\right)\right] \qquad (2.3)$$

2.4 Adjusting Nominal Effort

Cost drivers are used to capture characteristics of the software development that affect the effort to complete the project. Cost drivers that have a multiplicative effect on predicting effort are called Effort Multipliers (EM). Each EM has a rating level that expresses the impact of the multiplier on development effort, PM. These rating can range from Extra Low to Extra High. For the purposes of quantitative analysis, each rating level of each EM has a weight associated with it. The nominal or average weight for an EM is 1.0. If a rating level causes more software development effort, then its corresponding EM weight is above 1.0. Conversely, if the rating level reduces the effort then the corresponding EM weight is less than 1.0. The selection of effort-multipliers is based on a strong rationale that they would independently explain a significant source of project effort or productivity variation.

2.4.1 Early Design Model

The Early Design model is used in the early stages of a software project when very little may be known about the size of the product to be developed, the nature of the target platform, the nature of the personnel to be involved in the project, or the detailed specifics of the process to be used. This model could be employed in Application Generator, System Integration, or Infrastructure development sectors.

The Early Design model adjusts the nominal effort using 7 EMs, Equation 2.4. Each multiplier has 7 possible weights. The cost drivers for this model are explained in the next chapter.

$$PM_{adjusted} = PM_{no\min al} \times \left(\prod_{i=1}^{7} EM_i \right) \tag{2.4}$$

2.4.2 Post-Architecture Model

The Post-Architecture model is the most detailed estimation model and it is intended to be used when a software life-cycle architecture has been developed. This model is used in the development and maintenance of software products in the Application Generators, System Integration, or Infrastructure sectors, see Chapter 1.

The Post-Architecture model adjusts nominal effort using 17 effort multipliers. The larger number of multipliers takes advantage of the greater knowledge available later in the development stage. The Post-Architecture effort multipliers are explained in the next chapter

$$PM_{adjusted} = PM_{no\min al} \times \left(\prod_{i=1}^{17} EM_i \right) \tag{2.5}$$

2.5 Development Schedule Estimation

COCOMO II provides a simple schedule estimation capability. The baseline schedule equation for all three COCOMO II stages is:

$$TDEV = \left[3.67 \times (\overline{PM})^{(0.28+0.2\times(B-.91))} \right] \times \frac{SCED\%}{100} \tag{2.6}$$

Where *TDEV* is the calendar time in months from the determination of a product's requirements baseline to the completion of an acceptance activity certifying that the product satisfies its requirements. *PM* is the estimated person-months excluding the SCED effort multiplier, *B* is the sum of project scale factors (discussed in the next chapter) and SCED% is the compression / expansion percentage in the SCED effort multiplier in Table 3-14 and Appendix 3.

3 COCOMO II Model Detail

3.1 Determining Size

3.1.1 Lines of Code

In COCOMO II, the logical source statement has been chosen as the standard line of code. Defining a line of code is difficult due to conceptual differences involved in accounting for executable statements and data declarations in different languages. The goal is to measure the amount of intellectual work put into program development, but difficulties arise when trying to define consistent measures across different languages. Table 3.1 shows a portion of the Software Engineering Institute's (SEI) definition checklist as it is being applied to support the development of the COCOMO II model. Detailed information and the complete checklist are available [Model Manual].

Table 3.1 Definition Checklist for Source Statements Counts

Definition name: __Logical Source Statements___Date:________________
________________(basic definition)__________Originator:_COCOMO II____

Measurement unit:	Physical source lines	☐			
	Logical source statements	✓			
Statement type	Definition ✓	Data Array ☐		Includes	Excludes
When a line or statement contains more than one type, classify it as the type with the highest precedence.					
1 Executable		Order of precedence →	1	✓	
2 Nonexecutable					
3 Declarations			2	✓	
4 Compiler directives			3	✓	
5 Comments					
6 On their own lines			4		✓
7 On lines with source code			5		✓
8 Banners and non-blank spacers			6		✓
9 Blank (empty) comments			7		✓
10 Blank lines			8		✓
11					
12					
How produced	Definition ✓	Data array ☐		Includes	Excludes
1 Programmed				✓	
2 Generated with source code generators					✓
3 Converted with automated translators				✓	
4 Copied or reused without change				✓	
5 Modified				✓	
6 Removed					✓
7					
8					
Origin	Definition ✓	Data array ☐		Includes	Excludes
1 New work: no prior existence				✓	
2 Prior work: taken or adapted from					
3 A previous version, build, or release				✓	
4 Commercial, off-the-shelf software (COTS), other than libraries					✓
5 Government furnished software (GFS), other than reuse libraries					✓
6 Another product					✓
7 A vendor-supplied language support library (unmodified)					✓
8 A vendor-supplied operating system or utility (unmodified)					✓
9 A local or modified language support library or operating system					✓
10 Other commercial library					✓
11 A reuse library (software designed for reuse)				✓	
12 Other software component or library				✓	
13					
14					

3.1.2 Function Points

The function point cost estimation approach is based on the amount of functionality in a software project and a set of individual project factors. Function points measure a software project by quantifying the information processing functionality associated with major external data or control input, output, or file.

3.2 Breakage

COCOMO II uses a breakage percentage, BRAK, to adjust the effective size of the product. Breakage reflects the requirements volatility in a project. It is the percentage of code thrown away due to requirements changes. For example, a project which delivers 100,000 instructions but discards the equivalent of an additional 20,000 instructions has a BRAK value of 20. This would be used to adjust the project's effective size to 120,000 instructions for a COCOMO II estimation

3.3 Adjusting for Reuse

COCOMO adjusts for the reuse by modifying the size of the module or project. The model treats reuse with function points and source lines of code the same in either the Early Design model or the Post-Architecture model.

3.3.1 Nonlinear Reuse Effects

Analysis of reuse costs indicates that the reuse cost function is nonlinear in two significant ways (see Figure 3-1):

- It does not go through the origin. There is generally a cost of about 5% for assessing, selecting, and assimilating the reusable component.
- Small modifications generate disproportionately large costs. This is primarily due to two factors: the cost of understanding the software to be modified, and the relative cost of interface checking.

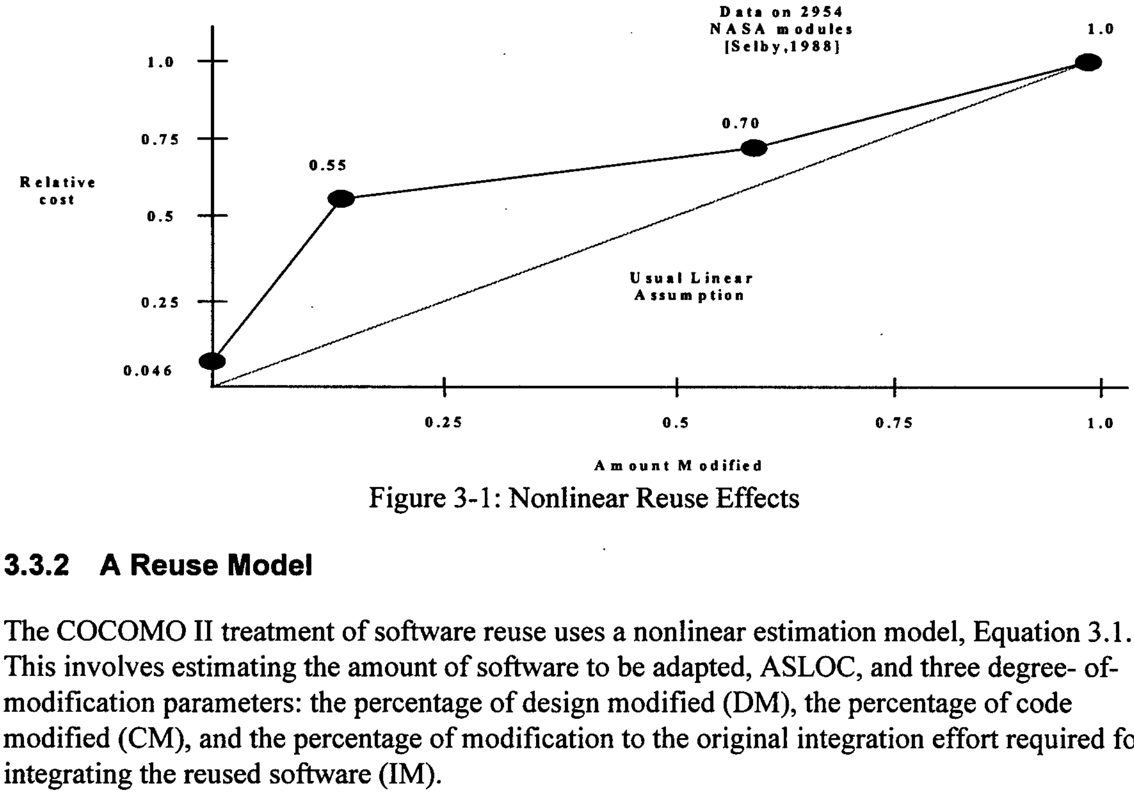


Figure 3-1: Nonlinear Reuse Effects

3.3.2 A Reuse Model

The COCOMO II treatment of software reuse uses a nonlinear estimation model, Equation 3.1. This involves estimating the amount of software to be adapted, ASLOC, and three degree- of-modification parameters: the percentage of design modified (DM), the percentage of code modified (CM), and the percentage of modification to the original integration effort required for integrating the reused software (IM).

The *Software Understanding* increment (SU) is obtained from Table 3-2. SU is expressed quantitatively as a percentage. If the software is rated very high on structure, applications clarity, and self-descriptiveness, the software understanding and interface checking penalty is 10%. If the software is rated very low on these factors, the penalty is 50%. SU is determined by taking the subjective average of the three categories.

Table 3-2: Rating Scale for Software Understanding Increment SU

	Very Low	Low	Nom	High	Very High
Structure	Very low cohesion, high coupling, spaghetti code.	Moderately low cohesion, high coupling.	Reasonably well structured; some weak areas.	High cohesion, low coupling.	Strong modularity, information hiding in data / control structures.
Application Clarity	No match between program and application world views.	Some correlation between program and application.	Moderate correlation between program and application.	Good correlation between program and application.	Clear match between program and application world-views.
Self-Descriptiveness	Obscure code; documentation missing, obscure or obsolete	Some code commentary and headers; some useful documentation.	Moderate level of code commentary, headers, documentations.	Good code commentary and headers; useful documentation; some weak areas.	Self-descriptive code; documentation up-to-date, well organized, with design rationale.
SU Increment to ESLOC	50	40	30	20	10

The other nonlinear reuse increment deals with the degree of *Assessment and Assimilation* (AA) needed to determine whether a fully reused software module is appropriate to the application, and to integrate its description into the overall product description. Table 3-3 provides the rating scale and values for the assessment and assimilation increment. AA is a percentage.

Table 3-3: Rating Scale for Assessment and Assimilation Increment (AA)

AA Increment	Level of AA Effort
0	None
2	Basic module search and documentation
4	Some module Test and Evaluation (T&E), documentation
6	Considerable module T&E, documentation
8	Extensive module T&E, documentation

The amount of effort required to modify existing software is a function not only of the amount of modification (AAF) and understandability of the existing software (SU), but also of the programmer's relative unfamiliarity with the software (UNFM). The UNFM parameter is applied multiplicatively to the software understanding effort increment. If the programmer works with the software every day, the 0.0 multiplier for UNFM will add no software understanding increment.

If the programmer has never seen the software before, the 1.0 multiplier will add the full software understanding effort increment. The rating of UNFM is in Table 3-4.

Table 3-4: Rating Scale for Programmer Unfamiliarity (UNFM)

UNFM Increment	Level of Unfamiliarity
0.0	Completely familiar
0.2	Mostly familiar
0.4	Considerably familiar
0.6	Somewhat familiar
0.8	Mostly unfamiliar
1.0	Completely unfamiliar

$$AAF = 0.4(DM) + 0.3(CM) + 0.3(IM)$$

$$ESLOC = \frac{ASLOC[AA + AAF(1 + 0.02(SU)(UNFM))]}{100}, AAF \leq 0.5 \qquad (3.1)$$

$$ESLOC = \frac{ASLOC[AA + AAF + (SU)(UNFM)]}{100}, AAF > 0.5$$

Equation 3.1 is used to determine an equivalent number of new instructions, equivalent source lines of code (ESLOC). ESLOC is divided by one thousand to derive KESLOC which is used as the COCOMO size parameter. The calculation of ESLOC is based on an intermediate quantity, the Adaptation Adjustment Factor (AAF). The adaptation quantities, DM, CM, IM are used to calculate AAF where:

- DM: Percent Design Modified. The percentage of the adapted software's design which is modified in order to adapt it to the new objectives and environment. (This is necessarily a subjective quantity.)
- CM: Percent Code Modified. The percentage of the adapted software's code which is modified in order to adapt it to the new objectives and environment.
- IM: Percent of Integration Required for Modified Software. The percentage of effort required to integrate the adapted software into an overall product and to test the resulting product as compared to the normal amount of integration and test effort for software of comparable size.

If there is no DM or CM (the component is being used unmodified) then there is no need for SU. If the code is being modified then SU applies.

3.4 Adjusting for Re-engineering or Conversion

The COCOMO II reuse model needs additional refinement to estimate the costs of software re-engineering and conversion. The major difference in re-engineering and conversion is the efficiency of automated tools for software restructuring. Equation 3.2 shows how automated translation affects the estimated nominal effort, *PM.*

$$PM_{nominal} = A \times (Size)^{B} + \left[\frac{ASLOC\left(\frac{AT}{100}\right)}{ATPROD} \right] \qquad (3.2)$$

3.5 Applications Maintenance

COCOMO II uses the reuse model for maintenance when the amount of added or changed base source code is less than or equal to 20% of the new code being developed. Base code is source code that already exists and is being changed for use in the current project. For maintenance projects that involve more than 20% change in the existing base code (relative to new code being developed) COCOMO II uses maintenance size. An initial maintenance size is obtained in one of two ways, Equation 3.3 or Equation 3.5. Equation 3.3 is used when the base code size is known and the percentage of change to the base code is known.

$$(SIZE)_M = [(Base\ Code\ Size) \cdot MCF] \cdot MAF \tag{3.3}$$

The percentage of change to the base code is called the Maintenance Change Factor (MCF). The MCF is for maintenance periods other than a year. Conceptually the MCF represents the ratio in Equation 3.4:

$$MCF = \frac{Size\ Added + Size\ Modified}{Base\ Code\ Size} \tag{3.4}$$

Equation 3.5 is used when the fraction of code added or modified to the existing base code during the maintenance period is known. Deleted code is not counted.

$$(Size)_M = (Size\ Added + Size\ Modified) \cdot MAF \tag{3.5}$$

The size can refer to thousands of source lines of code (KSLOC), Function Points, or Object Points. When using Function Points or Object Points, it is better to estimate MCF in terms of the fraction of the overall application being changed, rather than the fraction of inputs, outputs, screens, reports, etc. touched by the changes. Our experience indicates that counting the items touched can lead to significant over estimates, as relatively small changes can touch a relatively large number of items.

The initial maintenance size estimate (described above) is adjusted with a Maintenance Adjustment Factor (MAF), Equation 3.6. COCOMO II uses the Software Understanding (SU) and Programmer Unfamiliarity (UNFM) factors from its reuse model to model the effects of well or poorly structured/understandable software on maintenance effort.

$$MAF = 1 + \left(\frac{SU}{100} \cdot UNFM\right) \tag{3.6}$$

The resulting maintenance effort estimation formula is the same as the COCOMO II Post-Architecture development model:

$$PM_M = A \cdot (SIZE_M)^B \cdot \prod_{i=1}^{17} EM_i \tag{3.7}$$

The COCOMO II approach to estimating either the maintenance activity duration, T_M, or the average maintenance staffing level, FSP_M, is via the relationship:

$$PM_M = T_M \cdot FSP_M \tag{3.8}$$

Most maintenance is done as a level of effort activity. This relationship can estimate the level of effort, FSP_M, given T_M (as in annual maintenance estimates, where T_M = 12 months), or vice-versa (given a fixed maintenance staff level, FSP_M, determine the necessary time, T_M, to complete the effort).

3.6 Effort Multipliers

The application of project scale factors is the same for Early Design and the Post-Architecture models and was described in section 2.3. In the Early Design model a reduced set of the Post Architecture cost drivers are used.

3.6.1 Early Design

Appendix 2 contains the numeric values for the COCOMO II—1998 Effort Multiplers. The rest of this section discusses the scale factors in terms of their nominal levels.

The Early Design model uses KSLOC for size. Unadjusted function points are converted to the equivalent SLOC and then to KSLOC. The Early Design cost drivers are obtained by combining the Post-Architecture model cost drivers from Table 3-14. Whenever an assessment of a cost driver is between the rating levels always round to the Nominal rating, e.g. if a cost driver rating is between Very Low and Low, then select Low.

Table 3-5: Early Design and Post-Architecture Effort Multipliers

Early Design Cost Driver	Counterpart Combined Post-Architecture Cost Drivers
RCPX	RELY, DATA, CPLX, DOCU
RUSE	RUSE
PDIF	TIME, STOR, PVOL
PERS	ACAP, PCAP, PCON
PREX	AEXP, PEXP, LTEX
FCIL	TOOL, SITE
SCED	SCED

Overall Approach: Personnel Capability (PERS) Example

The following approach is used for mapping the full set of Post-Architecture cost drivers and rating scales onto their Early Design model counterparts. It involves the use and combination of numerical equivalents of the rating levels. Specifically, a Very Low Post-Architecture cost driver rating corresponds to a numerical rating of 1, Low is 2, Nominal is 3, High is 4, Very High is 5, and Extra High is 6. For the combined Early Design cost drivers, the numerical values of the contributing Post-Architecture cost drivers, Table 3-5, are summed, and the resulting totals are allocated to an expanded Early Design model rating scale going from Extra Low to Extra High. The Early Design model rating scales always have a Nominal total equal to the sum of the Nominal ratings of its contributing Post-Architecture elements.

Table 3-6: PERS Rating Levels

	Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Sum of ACAP, PCAP, PCON Ratings	3, 4	5, 6	7, 8	9	10, 11	12, 13	14, 15
Combined ACAP and PCAP Percentile	20%	39%	45%	55%	65%	75%	85%
Annual Personnel Turnover	45%	30%	20%	12%	9%	5%	4%

The rating scales and effort multipliers for PERS and the other Early Design cost drivers maintain consistent relationships with their Post-Architecture counterparts. For example, the PERS Extra Low rating levels (20% combined ACAP and PCAP percentile; 45% personnel turnover) represent averages of the ACAP, PCAP, and PCON rating levels adding up to 3 or 4.

Maintaining these consistency relationships between the Early Design and Post-Architecture rating levels ensures consistency of Early Design and Post-Architecture cost estimates. It also enables the rating scales for the individual Post-Architecture cost drivers, Table 3-14, to be used as detailed backups for the top-level Early Design rating scales given below.

Product Reliability and Complexity (RCPX)

This Early Design cost driver combines the four Post-Architecture cost drivers Required Software Reliability (RELY), Data size (DATA), Product complexity (CPLX), and Documentation match to life cycle needs (DOCU). Unlike the PERS components, the RCPX components have rating scales with differing width. RELY and DOCU range from Very Low to Very High; DATA ranges from Low to Very High, and CPLX ranges from Very Low to Extra High. The numerical sum of their ratings thus ranges from 5 (VL, L, VL, VL) to 21 (VH, VH, EH, VH).

Table 3-7 assigns RCPX ratings across this range, and associates appropriate rating scales to each of the RCPX ratings from Extra Low to Extra High. As with PERS, the Post- Architecture RELY, DATA, CPLX, and DOCU rating scales in Table 3-14 provide detailed backup for interpreting the Early Design RCPX rating levels.

Table 3-7: RCPX Rating Levels

	Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Sum of RELY, DATA, CPLX, DOCU Ratings	5, 6	7, 8	9 - 11	12	13 - 15	16 - 18	19 - 21
Emphasis on reliability, documentation	Very little	Little	Some	Basic	Strong	Very Strong	Extreme
Product complexity	Very simple	Simple	Some	Moderate	Complex	Very complex	Extremely complex
Data size	Small	Small	Small	Moderate	Large	Very Large	Very Large

Required Reuse (RUSE)

This Early Design model cost driver is the same as its Post- Architecture counterpart, which is covered in the section on the Post-Architecture model. A summary of its rating levels is given below and in Table 3-14.

Table 3-8: RUSE Rating Level Summary

	Very Low	Low	Nominal	High	Very High	Extra High
RUSE		none	across project	across program	across product line	across multiple product lines

Platform Difficulty (PDIF)

This Early Design cost driver combines the three Post- Architecture cost drivers execution time (TIME), main storage constraint (STOR), and platform volatility (PVOL). TIME and STOR range from Nominal to Extra High; PVOL ranges from Low to Very High. The numerical sum of their ratings thus ranges from 8 (N, N, L) to 17 (EH, EH, VH).

Table 3-9 assigns PDIF ratings across this range, and associates the appropriate rating scales to each of the PDIF rating levels. The Post-Architecture rating scales in Table 3-14 provide additional backup definition for the PDIF ratings levels.

Table 3-9: PDIF Rating Levels

	Low	Nominal	High	Very High	Extra High
Sum of TIME, STOR, and PVOL ratings	8	9	10 - 12	13 - 15	16, 17
Time and storage constraint	50%	50%	65%	80%	90%
Platform volatility	Very stable	Stable	Somewhat volatile	Volatile	Highly volatile

Personnel Experience (PREX)

This Early Design cost driver combines the three Post-Architecture cost drivers application experience (AEXP), platform experience (PEXP), and language and tool experience (LTEX). Each of these range from Very Low to Very High; as with PERS, the numerical sum of their ratings ranges from 3 to 15.

Table 3-10 assigns PREX ratings across this range, and associates appropriate effort multipliers and rating scales to each of the rating levels.

Table 3-10: PREX Rating Levels

	Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Sum of AEXP, PEXP, and LTEX ratings	3, 4	5, 6	7, 8	9	10, 11	12, 13	14, 15
Applications, Platform, Language and Tool Experience	□ 3 mo.	5 months	9 months	1 year	2 years	4 years	6 years

Facilities (FCIL)

This Early Design cost driver combines the two Post-Architecture cost drivers: use of software tools (TOOL) and multisite development (SITE). TOOL ranges from Very Low to Very High; SITE ranges from Very Low to Extra High. Thus, the numerical sum of their ratings ranges from 2 (VL, VL) to 11 (VH, EH).

Table 3-11 assigns FCIL ratings across this range, and associates appropriate rating scales to each of the FCIL rating levels. The individual Post-Architecture TOOL and SITE rating scales in Table 3-11 again provide additional backup definition for the FCIL rating levels.

Table 3-11: FCIL Rating Levels

	Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Sum of TOOL and SITE ratings	2	3	4, 5	6	7, 8	9, 10	11
TOOL support	Minimal	Some	Simple CASE tool collectio n	Basic life-cycle tools	Good; moder-ately inte-grated	Strong; moder-ately inte-grated	Strong; well integrated
Multisite conditions	Weak support of complex multisite develop-ment	Some support of complex M/S devel.	Some support of moder-ately complex M/S devel.	Basic support of moder-ately complex M/S devel.	Strong support of moder-ately complex M/S devel.	Strong support of simple M/ S devel.	Very strong support of collocated or simple M/S devel.

Schedule (SCED)

The Early Design cost driver is the same as its Post-Architecture counterpart. A summary of its rating levels is given in Table 3-12 below.

Table 3-12: SCED Rating Level Summary

	Very Low	Low	Nominal	High	Very High	Extra High
SCED	75% of nominal	85%	100%	130%	160%	

3.6.2 Post-Architecture

Appendix 3 contains the numeric values for the COCOMO II—1998 Effort Multiplers. The rest of the sections discusses the scale factors in terms of their nominal levels.

These are the 17 effort multipliers used in COCOMO II Post-Architecture model to adjust the nominal effort, Person Months, to reflect the software product under development. They are grouped into four categories: product, platform, personnel, and project. Figure 3-18 lists the different cost drivers with their rating criterion (found at the end of this section). Whenever an assessment of a cost driver is between the rating levels always round to the Nominal rating, e.g. if a cost driver rating is between High and Very High, then select High. The counterpart 7 effort multipliers for the Early Design model are discussed in the chapter explaining that model

3.6.2.1 Product Factors

Required Software Reliability (RELY)

This is the measure of the extent to which the software must perform its intended function over a period of time. If the effect of a software failure is only slight inconvenience then RELY is low. If a failure would risk human life then RELY is very high.

	Very Low	Low	Nominal	High	Very High	Extra High
RELY	slight inconvenience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	

Data Size (DATA)

This measure attempts to capture the affect large data requirements have on product development. The rating is determined by calculating D/P. The reason the amount of data is important to consider it because of the effort required to generate the test data that will be used to exercise the program.

$$\frac{D}{P} = \frac{Data\ Size\ (Bytes)}{Program\ Size\ (SLOC)} \qquad \text{EQ. 1}$$

DATA is rated as low if D/P is less than 10 and it is very high if it is greater than 1000.

	Very Low	Low	Nominal	High	Very High	Extra High
DATA		D bytes/ Pgm SLOC < 10	10 D/P < 100	100 D/P < 1000	D/P≥ 1000	

Product Complexity (CPLX)

Table 3-13 (found at the end of this section) provides the CPLX rating scale. Complexity is divided into five areas: control operations, computational operations, device-dependent operations, data management operations, and user interface management operations. Select the area or combination of areas that characterize the product or a sub-system of the product. The complexity rating is the subjective weighted average of these areas.

Table 3-13: Module Complexity Ratings versus Type of Module

	Control Operations	Computational Operations	Device-dependent Operations	Data Management Operations	User Interface Management Operations
Very Low	Straight-line code with a few non-nested structured programming operators: DOs, CASEs, IFTHENELSEs. Simple module composition via procedure calls or simple scripts.	Evaluation of simple expressions: e.g., _A=B+C*(D-E)_	Simple read, write statements with simple formats.	Simple arrays in main memory. Simple COTS-DB queries, updates.	Simple input forms, report generators.
Low	Straightforward nesting of structured programming operators. Mostly simple predicates	Evaluation of moderate-level expressions: e.g., _D=SQRT(B**2-4.*A*C)_	No cognizance needed of particular processor or I/O device characteristics. I/O done at GET/PUT level.	Single file subsetting with no data structure changes, no edits, no intermediate files. Moderately complex COTS-DB queries, updates.	Use of simple graphic user interface (GUI) builders.
Nominal	Mostly simple nesting. Some intermodule control. Decision tables. Simple callbacks or message passing, including middleware-supported distributed processing	Use of standard math and statistical routines. Basic matrix/vector operations.	I/O processing includes device selection, status checking and error processing.	Multi-file input and single file output. Simple structural changes, simple edits. Complex COTS-DB queries, updates.	Simple use of widget set.
High	Highly nested structured programming operators with many compound predicates. Queue and stack control. Homogeneous, distributed processing. Single processor soft real-time control.	Basic numerical analysis: multivariate interpolation, ordinary differential equations. Basic truncation, roundoff concerns.	Operations at physical I/O level (physical storage address translations; seeks, reads, etc.). Optimized I/O overlap.	Simple triggers activated by data stream contents. Complex data restructuring.	Widget set development and extension. Simple voice I/O, multimedia.
Very High	Reentrant and recursive coding. Fixed-priority interrupt handling. Task synchronization, complex callbacks, heterogeneous distributed processing. Single-processor hard real-time control.	Difficult but structured numerical analysis: near-singular matrix equations, partial differential equations. Simple parallelization.	Routines for interrupt diagnosis, servicing, masking. Communication line handling. Performance-intensive embedded systems.	Distributed database coordination. Complex triggers. Search optimization.	Moderately complex 2D/3D, dynamic graphics, multimedia.
Extra High	Multiple resource scheduling with dynamically changing priorities. Microcode-level control. Distributed hard real-time control.	Difficult and unstructured numerical analysis: highly accurate analysis of noisy, stochastic data. Complex parallelization.	Device timing-dependent coding, micro-programmed operations. Performance-critical embedded systems.	Highly coupled, dynamic relational and object structures. Natural language data management.	Complex multimedia, virtual reality.

Required Reusability (RUSE)

This cost driver accounts for the additional effort needed to construct components intended for reuse on the current or future projects. This effort is consumed with creating more generic design of software, more elaborate documentation, and more extensive testing to ensure components are ready for use in other applications.

	Very Low	Low	Nominal	High	Very High	Extra High
RUSE		none	across project	across pro gram	across product line	across multiple product lines

Documentation match to life-cycle needs (DOCU)

Several software cost models have a cost driver for the level of required documentation. In COCOMO II, the rating scale for the DOCU cost driver is evaluated in terms of the suitability of the project's documentation to its life-cycle needs. The rating scale goes from Very Low (many life-cycle needs uncovered) to Very High (very excessive for life-cycle needs).

	Very Low	Low	Nominal	High	Very High	Extra High
DOCU	Many life-cycle needs uncovered	Some life-cycle needs uncovered.	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	

3.6.2.2 Platform Factors

The platform refers to the target-machine complex of hardware and infrastructure software (previously called the virtual machine). The factors have been revised to reflect this as described in this section. Some additional platform factors were considered, such as distribution, parallelism, embeddedness, and real-time operations. These considerations have been accommodated by the expansion of the Module Complexity ratings in Equation 3-15.

Execution Time Constraint (TIME)

This is a measure of the execution time constraint imposed upon a software system. The rating is expressed in terms of the percentage of available execution time expected to be used by the system or subsystem consuming the execution time resource. The rating ranges from nominal, less than 50% of the execution time resource used, to extra high, 95% of the execution time resource is consumed.

	Very Low	Low	Nominal	High	Very High	Extra High
TIME			50% use of available execution time	70%	85%	95%

Main Storage Constraint (STOR)

This rating represents the degree of main storage constraint imposed on a software system or subsystem. Given the remarkable increase in available processor execution time and main storage, one can question whether these constraint variables are still relevant. However, many applications continue to expand to consume whatever resources are available, making these cost drivers still relevant. The rating ranges from nominal, less that 50%, to extra high, 95%.

	Very Low	Low	Nominal	High	Very High	Extra High
STOR			50% use of available storage	70%	85%	95%

Platform Volatility (PVOL)

"Platform" is used here to mean the complex of hardware and software (OS, DBMS, etc.) the software product calls on to perform its tasks. If the software to be developed is an operating system then the platform is the computer hardware. If a database management system is to be developed then the platform is the hardware and the operating system. If a network text browser is to be developed then the platform is the network, computer hardware, the operating system, and the distributed information repositories. The platform includes any compilers or assemblers supporting the development of the software system. This rating ranges from low, where there is a major change every 12 months, to very high, where there is a major change every two weeks.

	Very Low	Low	Nominal	High	Very High	Extra High
PVOL		major change every 12 mo.; minor change every 1 mo.	major: 6 mo.; minor: 2 wk.	major: 2 mo.; minor: 1 wk.	major: 2 wk.; minor: 2 days	

3.6.2.3 Personnel Factors

Analyst Capability (ACAP)

Analysts are personnel that work on requirements, high level design and detailed design. The major attributes that should be considered in this rating are Analysis and Design ability, efficiency and thoroughness, and the ability to communicate and cooperate. The rating should not consider the level of experience of the analyst; that is rated with AEXP. Analysts that fall in the 15th percentile are rated very low and those that fall in the 95th percentile are rated as very high.

	Very Low	Low	Nominal	High	Very High	Extra High
ACAP	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	

Programmer Capability (PCAP)

Current trends continue to emphasize the importance of highly capable analysts. However the increasing role of complex COTS packages, and the significant productivity leverage associated with programmers' ability to deal with these COTS packages, indicates a trend toward higher importance of programmer capability as well.

Evaluation should be based on the capability of the programmers as a team rather than as individuals. Major factors, which should be considered in the rating, are ability, efficiency and thoroughness, and the ability to communicate and cooperate. The experience of the programmer should not be considered here; it is rated with AEXP. A very low rated programmer team is in the 15th percentile and a very high rated programmer team is in the 95th percentile.

	Very Low	Low	Nominal	High	Very High	Extra High
PCAP	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	

Applications Experience (AEXP)

This rating is dependent on the level of applications experience of the project team developing the software system or subsystem. The ratings are defined in terms of the project team's equivalent level of experience with this type of application. A very low rating is for application experience of less than 2 months. A very high rating is for experience of 6 years or more.

	Very Low	Low	Nominal	High	Very High	Extra High
AEXP	2 months	6 months	1 year	3 years	6 years	

Platform Experience (PEXP)

The Post-Architecture model broadens the productivity influence of PEXP, recognizing the importance of understanding the use of more powerful platforms, including more graphic user interface, database, networking, and distributed middleware capabilities.

	Very Low	Low	Nominal	High	Very High	Extra High
PEXP	2 months	6 months	1 year	3 years	6 year	

Language and Tool Experience (LTEX)

This is a measure of the level of programming language and software tool experience of the project team developing the software system or subsystem. Software development includes the use of tools that perform requirements and design representation and analysis, configuration management, document extraction, library management, program style and formatting, consistency checking, etc. In addition to experience in programming with a specific language the supporting tool set also effects development time. A low rating is given for experience of less than 2 months. A very high rating is given for experience of 6 or more years.

	Very Low	Low	Nominal	High	Very High	Extra High
LTEX	2 months	6 months	1 year	3 years	6 year	

Personnel Continuity (PCON)

The rating scale for PCON is in terms of the project's annual personnel turnover: from 3%, very high, to 48%, very low.

	Very Low	Low	Nominal	High	Very High	Extra High
PCON	48% / year	24% / year	12% / year	6% / year	3% / year	

3.6.2.4 Project Factors

Use of Software Tools (TOOL)

Software tools have improved significantly since the 1970's projects used to calibrate COCOMO. The tool rating ranges from simple edit and code, very low, to integrated lifecycle management tools, very high.

	Very Low	Low	Nominal	High	Very High	Extra High
TOOL	edit, code, debug	simple, front end, back end CASE, little integration	basic life cycle tools, moderately integrated	strong, mature life cycle tools, moderately integrated	strong, mature, pro active life cycle tools, well integrated with processes, methods, reuse	

Multisite Development (SITE)

Given the increasing frequency of multisite developments, and indications that multisite development effects are significant, the SITE cost driver was added in COCOMO II. Determining its cost driver rating involves the assessment and averaging of two factors: site collocation (from fully collocated to international distribution) and communication support (from surface mail and some phone access to full interactive multimedia).

	Very Low	Low	Nominal	High	Very High	Extra High
SITE: Communications	Some phone, mail	Individual phone, FAX	Narrowband email	Wideband electronic communicati on.	Wideband elect. comm, occasional video conf.	Interactive multimedia

Required Development Schedule (SCED)

This rating measures the schedule constraint imposed on the project team developing the software. The ratings are defined in terms of the percentage of schedule stretch-out or acceleration with respect to a nominal schedule for a project requiring a given amount of effort. Accelerated schedules tend to produce more effort in the later phases of development because more issues are left to be determined due to lack of time to resolve them earlier. A schedule compress of 74% is rated very low. A stretch-out of a schedule produces more effort in the earlier phases of development where there is more time for thorough planning, specification and validation. A stretch-out of 160% is rated very high.

	Very Low	Low	Nominal	High	Very High	Extra High
SCED	75% of nominal	85%	100%	130%	160%	

Table 3-14: Post-Architecture Cost Driver Rating Level Summary

	Very Low	Low	Nominal	High	Very High	Extra High
RELY	slight inconvenience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	
DATA		DB bytes/Pgm SLOC < 10	10 D/P < 100	100 D/P < 1000	D/P 1000	
CPLX	see Table 3-13					
RUSE		none	across project	across program	across product line	across multiple product lines
DOCU	Many life-cycle needs uncovered	Some life-cycle needs uncovered.	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life- cycle needs	
TIME			50% use of available execution time	70%	85%	95%
STOR			50% use of available storage	70%	85%	95%
PVOL		major change every 12 mo.; minor change every 1 mo.	major: 6 mo.; minor: 2 wk.	major: 2 mo.; minor: 1 wk.	major: 2 wk.; minor: 2 days	
ACAP	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
PCAP	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
PCON	48% / year	24% / year	12% / year	6% / year	3% / year	
AEXP	2 months	6 months	1 year	3 years	6 years	
PEXP	2 months	6 months	1 year	3 years	6 year	
LTEX	2 months	6 months	1 year	3 years	6 year	
TOOL	edit, code, debug	simple, frontend, backend CASE, little integration	basic lifecycle tools, moderately integrated	strong, mature lifecycle tools, moderately integrated	strong, mature, proactive life cycle tools, well integrated with processes, methods, reuse	
SITE: Collocation	International	Multi-city and Multi-company	Multi-city or Multi-company	Same city or metro. area	Same building or complex	Fully collocated
SITE: Communications	Some phone, mail	Individual phone, FAX	Narrowband email	Wideband electronic communication.	Wideband elect. comm, occasional video conf.	Interactive multimedia
SCED	75% of nominal	85%	100%	130%	160%	

4.0 Numerical Values of Scale Factors for Early Design and Post-Architecture

A-Posteriori Bayesian Values (1998)

Scale Factors

Driver	VL	L	N	H	VH	XH
PREC	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	5.07	4.05	3.04	2.03	1.01	0.00
RESL	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	7.80	6.24	4.68	3.12	1.56	0.00

Equations

$$B = .91 + 0.01 \times \sum W_i \tag{2.2}$$

$$TDEV = \left[3.67 \times (\overline{PM})^{(0.28+0.2\times(B-.91))}\right] \times \frac{SCED\%}{100} \tag{2.6}$$

$$PM_{nominal} = 2.94 \cdot SIZE^{B} \cdot \prod_{i=1}^{17} PA - EM_i$$

-or-

$$PM_{nominal} = 2.94 \cdot SIZE^{B} \cdot \prod_{i=1}^{5} ED - EM_i$$

5.0 Numerical Values of Effort Multipliers for Early Design

From sdevnani@sunset.usc.edu Sun Aug 30 23:20:18 1998
Date: Mon, 17 Aug 1998 17:20:14 -0700
From: Sunita Chulani <sdevnani@sunset.usc.edu>
To: awbrown@sunset.usc.edu, Yu-Ting Kao <yutingk@scf.usc.edu>
Subject: Early Design Multipliers

Here are the Early Design Parameters

Effort Multipliers

Driver	XL	VL	L	N	H	VH	XH
PERS	2.12	1.62	1.26	1.00	0.83	0.63	.50
RCPX	0.73	0.81	0.98	1.00	1.30	1.74	2.38
PDIF		0.87	1.00	1.29	1.81	2.61	
PREX	1.59	1.33	1.12	1.00	0.87	0.71	0.62
FCIL	1.43	1.30	1.10	1.00	0.87	0.73	0.62

Equations

$$B = .91 + 0.01 \times \sum W_i \qquad (2.2)$$

$$TDEV = \left[3.67 \times (\overline{PM})^{(0.28+0.2\times(B-.91))}\right] \times \frac{SCED\%}{100} \qquad (2.6)$$

$$PM_{nominal} = 2.94 \cdot SIZE^{B} \cdot \prod_{i=1}^{5} ED - EM_i \text{-------}$$

the scale factor ratings stay the same.

- Sunita

6.0 Post-Architecture Scale Factors and Effort Multipliers

From sdevnani@sunset.usc.edu Mon Jun 29 09:29:09 1998

Dr. Horowitz and Jongmoon,

Here are the official COCOMO II.1998 values that need to be incorporated in the USC COOCMO II.1998 tool

A-Posteriori Bayesian Values (1998)

Scale Factors

Driver	VL	L	N	H	VH	XH
PREC	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	5.07	4.05	3.04	2.03	1.01	0.00
RESL	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	7.80	6.24	4.68	3.12	1.56	0.00

Effort Multipliers

Driver	VL	L	N	H	VH	XH
RELY	0.82	0.92	1.00	1.10	1.26	
DATA		0.90	1.00	1.14	1.28	
CPLX	0.73	0.87	1.00	1.17	1.34	1.74
RUSE		0.95	1.00	1.07	1.15	1.24
DOCU	0.81	0.91	1.00	1.11	1.23	
TIME			1.00	1.11	1.29	1.63
STOR			1.00	1.05	1.17	1.46
PVOL		0.87	1.00	1.15	1.30	
ACAP	1.42	1.19	1.00	0.85	0.71	
PCAP	1.34	1.15	1.00	0.88	0.76	
PCON	1.29	1.12	1.00	0.90	0.81	
AEXP	1.22	1.10	1.00	0.88	0.81	
PEXP	1.19	1.09	1.00	0.91	0.85	
LTEX	1.20	1.09	1.00	0.91	0.84	
TOOL	1.17	1.09	1.00	0.90	0.78	
SITE	1.22	1.09	1.00	0.93	0.86	0.80
SCED	1.43	1.14	1.00	1.00	1.00	

Equations

$$B = .91 + 0.01 \times \sum W_i \qquad (2.2)$$

$$TDEV = \left[3.67 \times (\overline{PM})^{(0.28+0.2\times(B-.91))}\right] \times \frac{SCED\%}{100} \qquad (2.6)$$

$$PM_{nominal} = 2.94 \cdot SIZE^{B} \cdot \prod_{i=1}^{17} PA-EM_i$$

For Effort : Multiplicative Constant = 2.94 and Baseline Exponent = 0.91
(This replaces 1.01 in the COCOMO II.1997 calibraiton)
For Schedule : Multiplicative Constant = 3.67 and Baseline Exponent = 0.28
(this replaces 0.33 in the COCOMO II.1997 Calibration)

References:

[Banker et al 1994a]. Banker, R. D. ; Hishui Chang; Kemper, C.F. "Evidence of Economies of Scale in Software Development"; Information and Software Technology Vol 36 # 5 pp 275-82, 1994

[Boehm 1989, pp. 436-37] Software Risk Management by Barry W. Boehm:IEEE Computer Society Press, Washington, DC

[Boehm and Royce 1989; Figures 4 and 5]. Proceedings 1989 COCOMO Users Group Meeting

[Paulk et al. 1993, 1993a]. Paulk, M.C.; Curtis, B.; Chrissis, M.B.; Weber, C.V. Capability maturity model, version 1.1 ; IEEE Software , Volume: 10 4 , Page(s): 18 -27

[Model Manual] COCOMO II Model Definition Manual can be downloaded at http://sunset.usc.edu/COCOMOII/cocomo.html#downloads

L. Appendix 2.

Part 1

CORADMO Summary

A. Winsor Brown

Abstract

The **CO***COMO* **RAD MO***DEL (CORADMO) is currently implemented in two parts: a front end staged schedule and effort model,* **CO***COMO* **S***taged* **S***chedule and* **E***ffort* **MO***DEL (COSSEMO), and a back end RAD model. COSSEMO's uses a different schedule estimation calculation than COCOMO II's simple one: COSSEMO's schedule estimation uses a more complex calculation for the low effort situations, those below 64 person-months. At this time there are no other COSSEMO "drivers" besides COCOMO II's calculated effort. The RAD model has its roots in the results of a 1997 CSE Focused Workshop on Rapid Application Development*[1]*. RAD is taken to mean application of any of a number of techniques or strategies to reduce software development cycle time. Five classes of strategies whose degree of implementation can be used to parameterize a schedule estimate given an effort estimate produced by COCOMOII-1998 were derived from the Focused Workshop's results. These strategies, which are over and above just adding people to the task, include development process re-engineering (DPRS), re-use and very high level languages (RVHL), collaboration efficiency (CLAB), architecture investment and risk Resolution (RESL), and pre-positioning of assets (PPOS).*

[1] B. Boehm, S. Chulani, and A. Egyed, "Knowledge Summary: USC-CSE Focused Workshop on Rapid Application Development," USC-CSE Technical Report, June 1997.

1. Introduction

The evolution of CORADMO and its companion/pre-processor model COSSEMO has its roots in several activities undertaken by the Center for Software Engineering: COCOMO-II, and a Rapid Application Development Focused Workshop.

1.1. Another step in the evolution of COCOMO-II

The COCOMO-II Model Manual provides the primary motive for this extension of COCOMO-II. "As COCOMO II evolves, it will have a more extensive schedule estimation model, reflecting the different classes of process model a project can use; the effects of reusable and COTS software; and the effects of applications composition capabilities."

1.2. COCOMO II Schedule

The COCOMO-II schedule, as presently implemented (COCOMO-II1998) reflects a waterfall process model, and not any of the currently accepted alternatives such as iterative, spiral or evolutionary. In addition, it has been observed that the COCOMO-II's duration calculation seems unreasonable for small projects, those with effort under two person years. Obviously, COCOMO-II does not address any of the Rapid Application Development (RAD) strategies that are being employed to reduce schedule and sometimes effort as well.

1.3. COCOMO-II Constructive Staged Schedule & Effort Model and Constructive RAD Schedule Estimation Model

In an effort to overcome these shortfalls, two extensions have been developed: the COCOMO-II Staged Schedule & Effort Model (COSSEMO) and the Constructive RAD schedule estimation Model CoRADMo.

2. Improving the Classic CoCoMo Model for Schedule

The classic CoCoMo model has deficiencies in several areas: a waterfall predilection, no drivers reflecting modern schedule reduction efforts, and small-effort projects.

2.1. New Drivers

In CSE's Focussed Workshop #9 on RAD, a RAD Opportunity Tree of strategies was presented. The strategies included some techniques that were already covered by the drivers of COCOMO-II as well as several that were not. An analysis of these new drivers produced a set of five drivers that reflect identifiable behavioral characteristics. These were

1. Reuse and Very High-level Languages (RVHL)
2. Development Process Reengineering (DPRS)
3. Collaboration Efficiency (CLAB)
4. Architecture, Risk Resolution (RESL)
5. Prepositioning Assets (PPOS)

These new drivers are reflected in the annotated "RAD Opportunity Tree " shown in Figure 1.

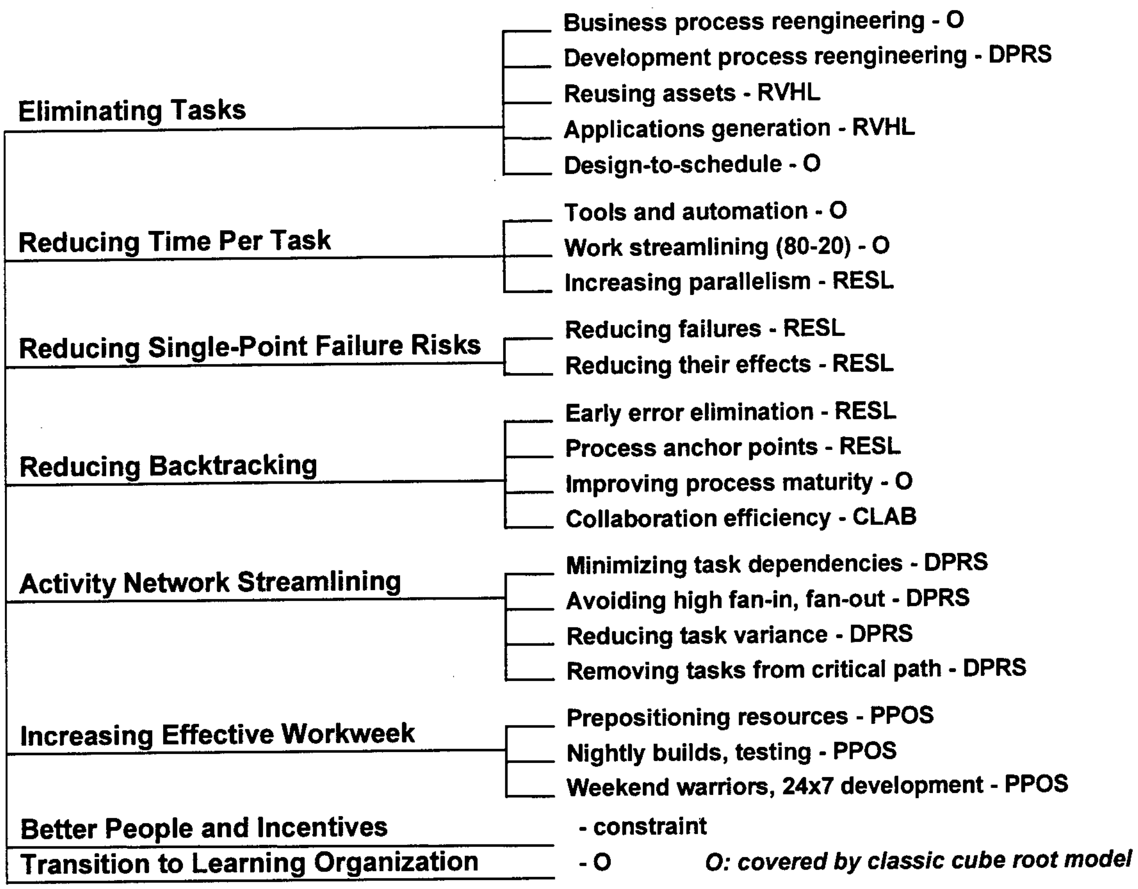

Eliminating Tasks
Business process reengineering - O
Development process reengineering - DPRS
Reusing assets - RVHL
Applications generation - RVHL
Design-to-schedule - O
Reducing Time Per Task
Tools and automation - O
Work streamlining (80-20) - O
Increasing parallelism - RESL
Reducing Single-Point Failure Risks
Reducing failures - RESL
Reducing their effects - RESL
Reducing Backtracking
Early error elimination - RESL
Process anchor points - RESL
Improving process maturity - O
Collaboration efficiency - CLAB
Activity Network Streamlining
Minimizing task dependencies - DPRS
Avoiding high fan-in, fan-out - DPRS
Reducing task variance - DPRS
Removing tasks from critical path - DPRS
Increasing Effective Workweek
Prepositioning resources - PPOS
Nightly builds, testing - PPOS
Weekend warriors, 24x7 development - PPOS
Better People and Incentives
- constraint
Transition to Learning Organization
- O
O: covered by classic cube root model

Figure 1. Annotated RAD Opportunity Tree

2.2. Duration Calculation

The COCOMO-II schedule, as presently implemented (in COCOMO-II1998) reflects a waterfall process model and its duration calculation seems unreasonable for small projects, those with effort under two person years.

2.2.1 COCOMO II Duration Calculation

The COCOMO-II duration calculation is based on an equation that has demonstrated historical accuracy, at least for large projects.

$$\textbf{Months} \sim 3\sqrt[3]{\textbf{Person-Months}}$$

This model component completely breaks down at very low efforts (16 person-months of effort) and is very questionable below a few person-years of effort.

2.2.2 COSSEMO Duration Calculation

COCOMO's effort and schedule estimates are focused on Elaboration and Construction (the Stages between LCO and IOC. Inception corresponds to the COCOMO's "Requirements" activity, which is actually an additional (fixed percentage) effort, above and beyond the effort calculated by COCOMO.

equation is

TDEV= (3.0 * PMbar^(0.33 + 0.2 * (B-1.01)) * SCED%/100

where *TDEV* is the calendar time in months from the determination of a product's requirements baseline to the completion of an acceptance activity certifying that the product satisfies its requirements. *PMbar* is the estimated person-months excluding the SCED effort multiplier, *B* is the sum of project scale factors (discussed in the next chapter) and SCED% is the compression / expansion percentage in the SCED effort multiplier.

The TDEV calculations mean that the calculated schedule is related, approximately, to three times the cube root of the effort. For low-effort situations, especially below twenty seven (27) person months, this yields a very pessimistic and unlikely duration of nine (9) months applying three (3) FSP people. As a result, a new baseline schedule equation for efforts below 16 months has been chosen which is based on the square-root of the effort, yielding equal FSPs and schedule months. A linear interpolation is used between the high-end applicability of 64 person months (which corresponds to a schedule of 14.4 months for a 100Ksloc EHART using 1998 average driver values), and the low end point of 16 person months.

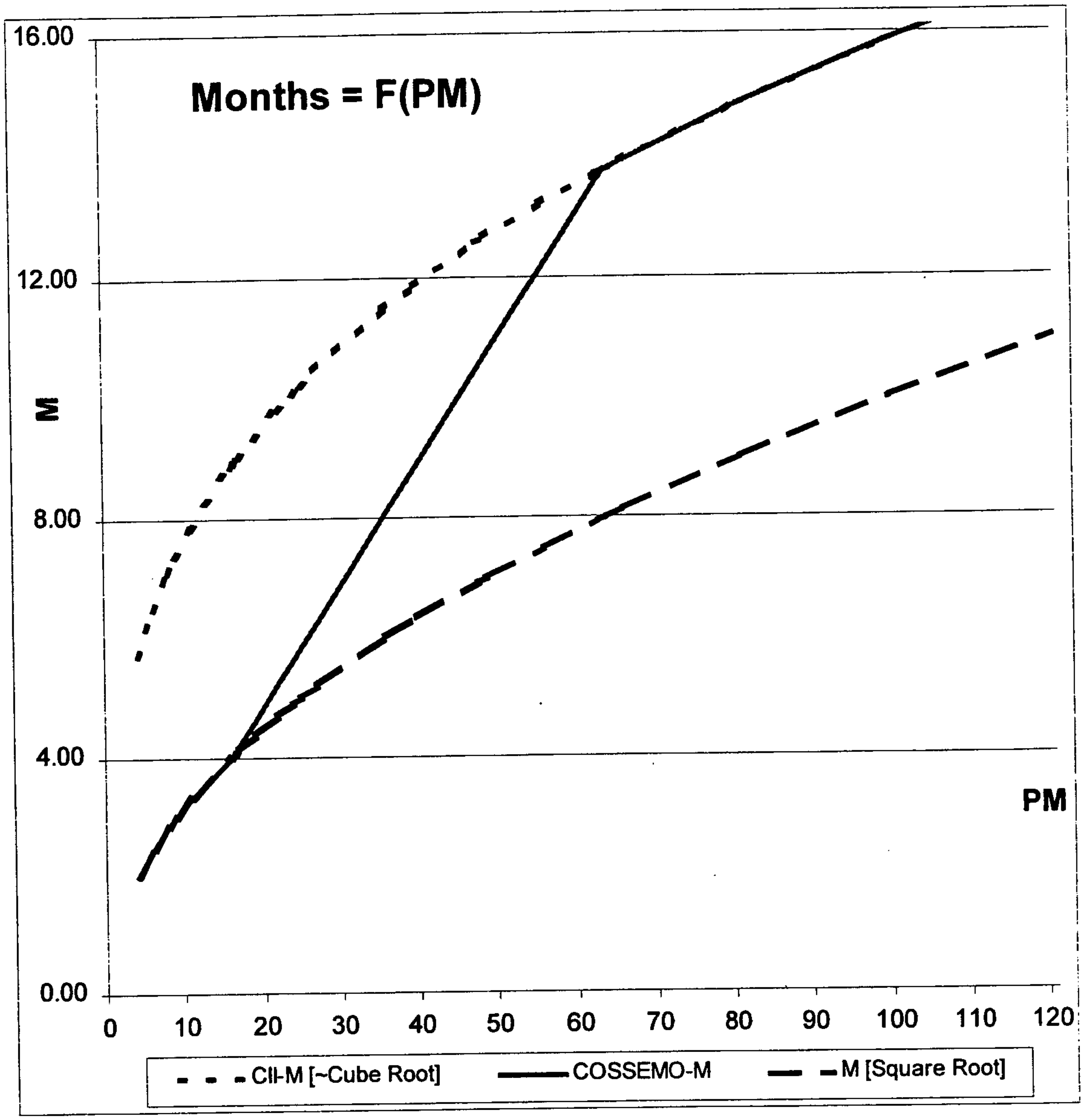


Figure 2. Showing linear interpolation between cube and square root

2.3. Process Model

The COSSEMO model is based on the lifecycle anchoring concepts discussed by Boehm[2]. The anchor points are defined as Life Cycle Objectives (LCO), Life Cycle Architecture (LCA), and Initial Operational Capability (IOC). An augmented illustration based on one from the Rational Corporation[3], Figure 3 shows the stages around the anchor points.

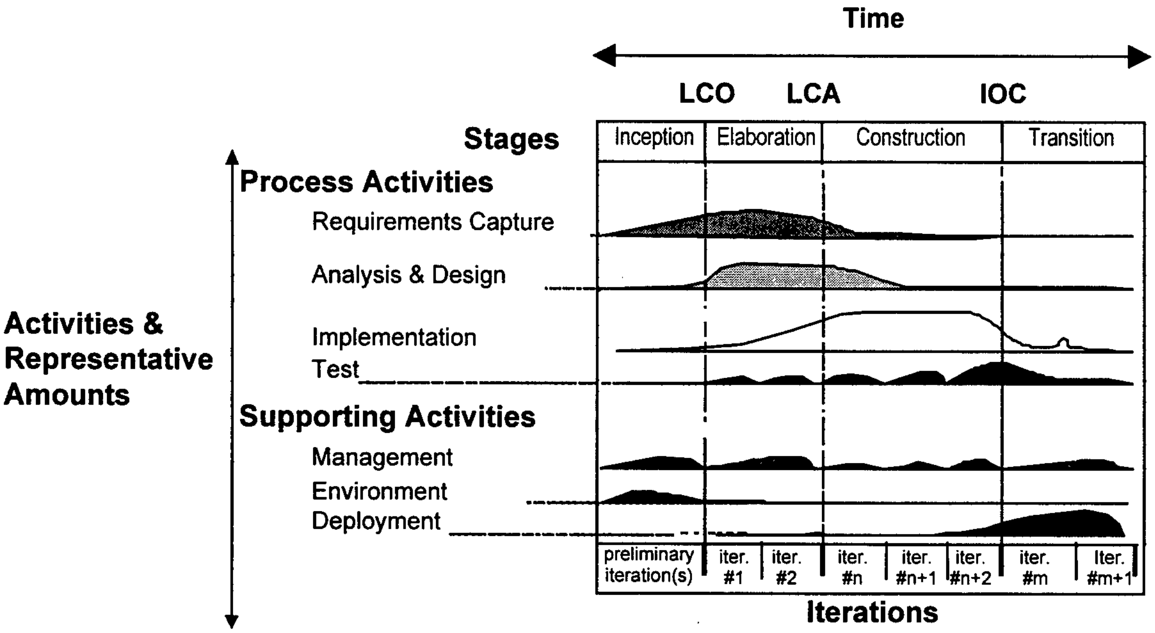


Figure 3. A modern lifecycle model with anchor points

[2] Barry W. Boehm, "Anchoring the Software Process," *IEEE Software,* 13, 4, July 1996, pp. 73-82.

[3] Rational Corp., "Rational Objectory Process 4.1 – Your UML Process", available at http://www.rational.com/support/techpapers/toratobjprcs/.

2.4. Anchor Points, Stages and Activities

The diagram shows various activities, and implies iterations and the relative effort and duration of typical cycles within an iteration. The following table provides some more detail on the relative proportion of the activities, and some details.

COCOMO II Submodel Usage	Early Design		Post-Architecture	Maintenance
	LCO	LCA	IOC	
Activities \ Stage	**Inception**	**Elaboration**	**Construction**	**Transition**
Requirements Capture	Some usually	Most, peaks here	Minor	None
Analysis & Design	A little	Majority, mostly constant effort	Some	Some, for repair during ODT&E
Implementation	Practically none	Some, usually for risk reduction	Bulk; mostly constant effort	Some, for repair during ODT&E
Test	None	Some, for prototypes	Most for unit test, integration test and qualification test.	Some, for repaired code.

Table 1. Stages, Anchor Points, and relative amount and kind of Activities

3. Model Overview

There are two parts of the current model, COSSEMO and CORADMO. They both assume that data is available from a COCOMO II model.

3.1. COCOMO II Constructive Staged Schedule & Effort Model (COSSEMO)

The COSSEMO part of the model currently has no drivers, per se. The model does allow for the specification of the percentages of effort and schedule to be applied to the different stages: Inception, Elaboration and Construction. The predicted effort and schedule from a COCOMO II run correspond to the sum of the Elaboration and Construction stages' effort and schedule, respectively. The percentages of effort and schedule Elaboration and Construction stages thus total 100% and are used to distribute the sum accordingly. The percentages of effort and schedule for the Inception stage are also applied to the COCOMO II run's effort and schedule, respectively. Thus, the sum of the effort or schedule for three stages can actually total more than 100% of the COCOMO II run's effort and schedule.

3.2. Constructive RAD Schedule Estimation Model (CORADMO)

The CORADMO model has five drivers. Each driver has both rating levels, which are selected by a user based on the characteristics of the software project, its development organization, and its milieu. There are numeric schedule and effort multiplier values per stage for each rating level. The rating levels are described in detail in Part 2 of this report, which corresponds to a subset of the information gathering worksheet for users of the model and its tools. The rating levels and their corresponding numerical values are summarized below and provided in full detail in Part 3 of this report.

3.2.1 Reuse and VHLLs (RVHL)

The impact of re-use of 3GL production code is handled directly in the COCOMO II model via the re-use sub-model and its effect on size. This CORADMO driver reflects the impact of re-use of code (other than production code) and/or the use of very high level languages, especially during the Inception and Elaboration stages. Higher rating levels reflect the potential schedule compression impacts in Inception and Elaboration stages due to faster prototyping, option exploration. Clearly this impact will be dependent on the level of capability and experience in doing this, such as Rapid Prototyping experience. The values of the multipliers corresponding to the rating levels are the same for both effort and schedule; this implies that the staff level (number of full time software personnel) is held constant.

3.2.2 Development Process Reengineering and Streamlining (DPRS)

The schedule impact of this driver reflects the inverse of the level of bureaucracy in which the developers must operate. More succinctly stated, this driver captures the degree to which the project and organization allow and encourage streamlined or re-engineered development processes. A detailed rating level scale is provided for this driver (see Part 3 of this report). The values of the multipliers corresponding to the rating levels are the same for both effort and schedule; this implies that the staff level (number of full time software personnel) is held constant.

3.2.3 Collaboration Efficiency (CLAB)

Teams and team members who can collaborate effectively can reduce both effort and schedule; those that don't collaborate effectively have increased schedule and effort (due to wasted time). Rather than invent a new behavioral characteristic, this driver's rating level is primarily determined by an appropriate combination of COCOMO II Post-Architecture SITE and TEAM driver ratings and the PREX Early Design driver ratings. The SITE rating needs to be augmented by the team's collaboration tool maturity and experience. The effects of collaboration tools are expected to help in domain analysis, option analysis, and negotiation. A detailed rating level process and scale is provided for this driver (see Part 3 of this report). The values of the multipliers corresponding to the rating levels are the same for both effort and schedule; this implies that the staff level (number of full time software personnel) is held constant.

3.2.4 Architecture / Risk Resolution (RESL)

The COCOMO II Architecture / Risk Resolution driver (RESL) enables parallel construction activities without the COCOMO II assumed effect of increased integration and testing costs. There is not any impact on the effort or schedule in the Inception and Elaboration stages. There is no change in effort because of RESL, only potential for schedule compression at higher RESL ratings. For this driver to be effective, it is assumed that a higher level of staffing is available and used during construction. Thus the multipliers corresponding to the rating levels are not the same for both effort and schedule.

3.2.5 Prepositioning Assets (PPOS)

This driver reflects the degree to which assets are pre-tailored to a project or physically pre-positioned and furnished to the project for use on demand. The assets include skilled or particularly knowledgeable, people's skill-level increases, and pro-active team-building. The assets that are being pre-positioned also include processes and tools, and architecture and componentry. In order to take advantage of PPOS, the organization must either be taking a product-line approach or have made a 3, 6 or 10% pre-Inception effort investment! PPOS multipliers reflect the increased effort associated with the pre-positioning activities as well as the corresponding decrease in schedule and increased personnel required.

4. Implementation Models

There are four implementations of the CORADMO/COSSEMO model at this time. The logical implementation model shows how the various drivers and models interrelate. The physical implementation model shows how the logical implementation model has been realized in spreadsheets, both the standalone spreadsheet extension and the multiple parallel version that is part of the Technology Impact Analyzer. The first three of these models are shown below. The fourth implementation model is described in detail in the Volume 1 of the KBSA Report.

4.1. Logical COCOMO II RAD Extension

Figure 4 shows a conceptual logical block diagram for implementation of the RAD Model. It assumes that the regular COCOMO II implementation is extended with stage distributions which are potential driven by language level (e.g., 3GL or 4GL), experience, etc. The output of COCOMO II is used as a baseline for effort and schedule by the RAD Extension. The stage distributions extension allocates the baseline effort and schedule by stage. The RAD extension itself is controlled by the five drivers (discussed in section 3), resulting in the RAD effort and schedule by stage.

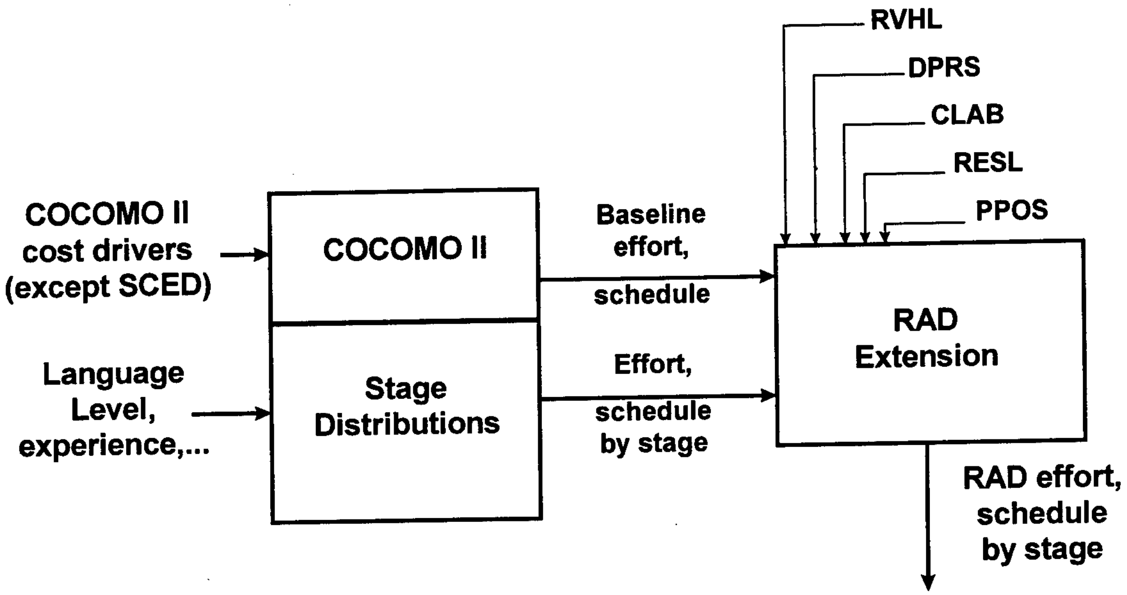

RVHL
DPRS
CLAB
RESL
PPOS
COCOMO II cost drivers (except SCED)
COCOMO II
Language Level, experience,...
Stage Distributions
Baseline effort, schedule
Effort, schedule by stage
RAD Extension
RAD effort, schedule by stage

Figure 4. Logical Implementation Model

4.2. Physical COCOMO II RAD Extension

Figure 5 shows the shows the current implementation strategy for the COCOMO II RAD extension. The upper leftt box represents the COCOMO II.98 model as implemented by COCOMO.exe, self-identified as "COCOMO II.1998.0" in its "About USC-COCOMOII" dialog box. Also part of the COCOMO II implementation suite is a spreadsheet called COCOMO.xls which is designed to import two CSV files that can be exported from COCOMO.exe and make their information available in spread sheet form (it also generates many useful charts and graphs of the data). The baseline effort and schedule as well as the values for all the drivers are acquired the COSSEMO Extension by links to the COCOMO.xls spreadsheet. The COSSEMO Extension, which is actually implemented as part of the RAD extension (CoRADMo.xls) distributes the effort (with no SCED impact) and schedule for subsequent operation by the RAD extension proper. Only the five new RAD drivers need to be input into the RAD extension: RESL is actually acquired from the COCOMO.xls spreadsheet via links, although that value can be over-ridden by the user.

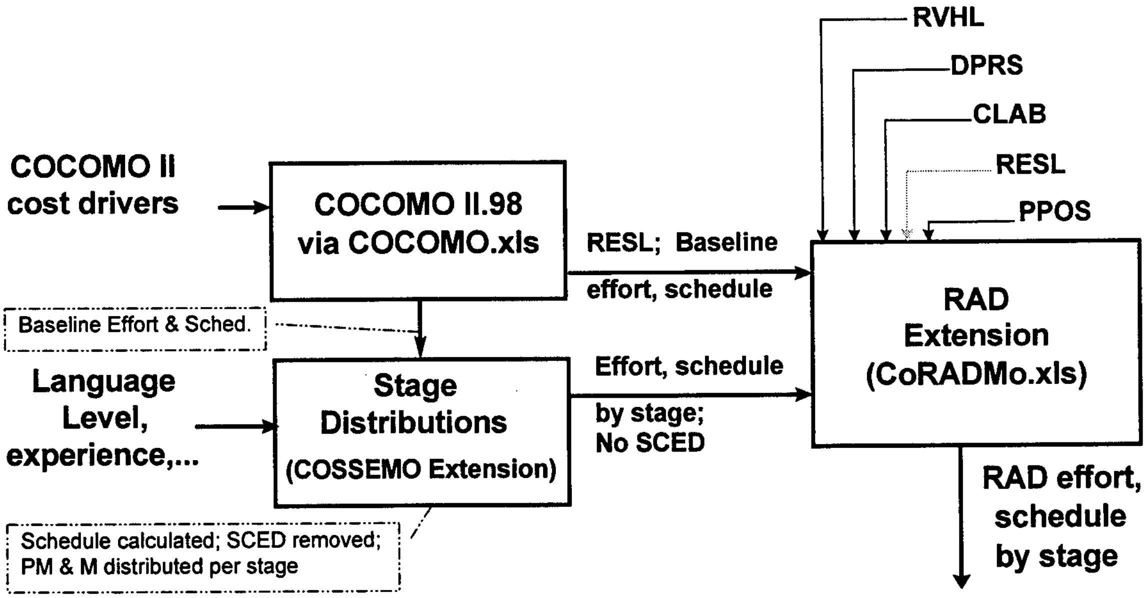
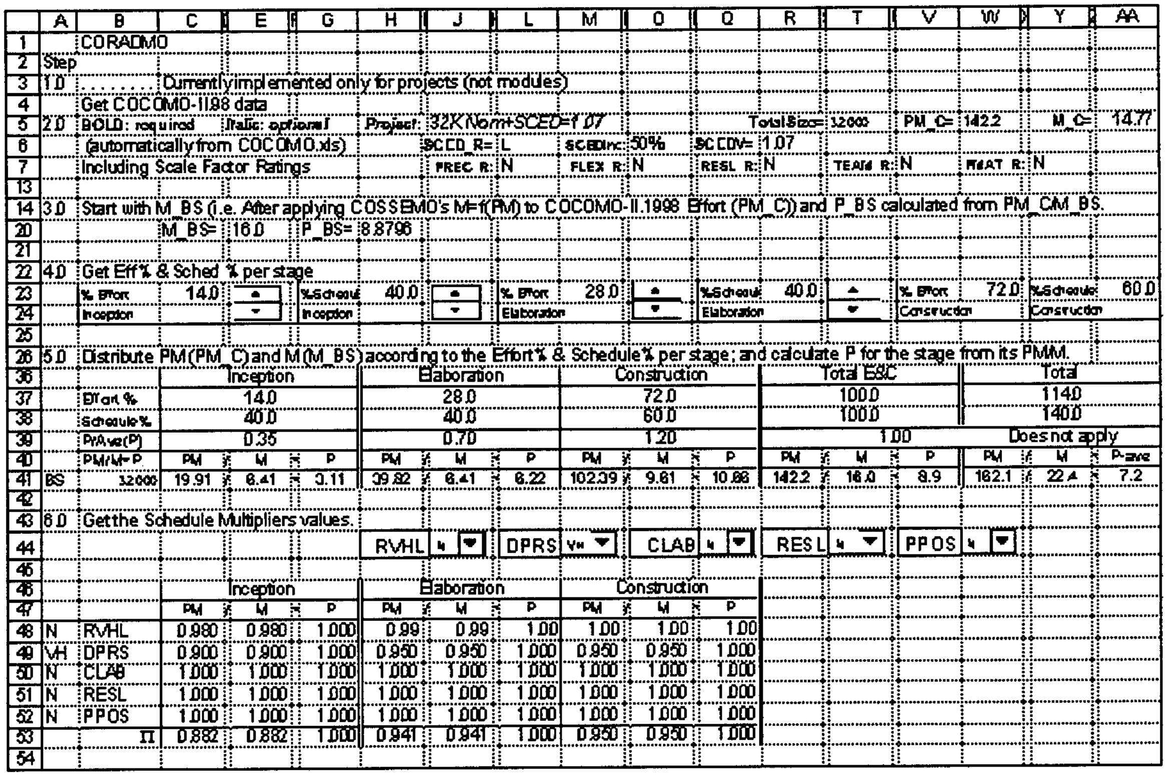

COCOMO II cost drivers
COCOMO II.98 via COCOMO.xls
Baseline Effort & Sched.
Language Level, experience,...
Stage Distributions (COSSEMO Extension)
Schedule calculated; SCED removed; PM & M distributed per stage
RESL; Baseline effort, schedule
Effort, schedule by stage; No SCED
RVHL
DPRS
CLAB
RESL
PPOS
RAD Extension (CoRADMo.xls)
RAD effort, schedule by stage

Figure 5. Physical Implementation Model

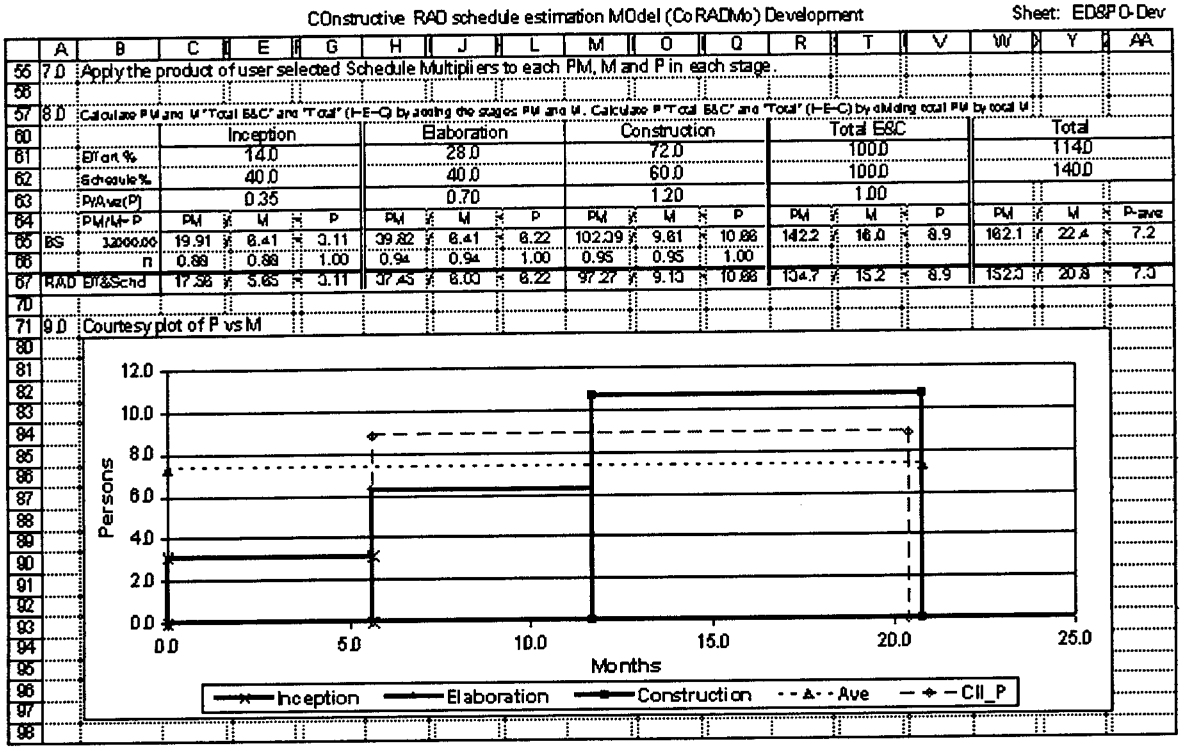
4.3. Stand-alone Spreadsheet Implementation

Figure and Figure 2 contain a stand-alone implementation of the COSSEMO and CORADMO extensions.



	A	B	C	E	G	H	J	L	M	O	Q	R	T	V	W	Y	AA
1		CORADMO															
2	Step																
3	1.0	....	Currently implemented only for projects (not modules)														
4		Get COCOMO-II.98 data															
5	2.0	BOLD: required		*Italic: optional*		*Project:*	*32K Nom+SCED=1.07*					TotalSize=	32000	PM_C=	142.2	M_C=	14.77
6		(automatically from COCOMO.xls)					SCED_R=	L	SCEDInc:	50%	SCCDV=	1.07					
7		Including Scale Factor Ratings					PREC R:	N	FLEX R:	N	RESL R:	N	TEAM R:	N	PMAT R:	N	
13																	
14	3.0	Start with M_BS (i.e. After applying COSSEMO's M=f(PM) to COCOMO-II.1998 Effort (PM_C)) and P_BS calculated from PM_C/M_BS.															
20			M_BS=	16.0	P_BS=	8.8796											
21																	
22	4.0	Get Eff% & Sched % per stage															
23		% Effort	14.0	▲	%Schedule	40.0	▲	% Effort	28.0	▲	%Schedule	40.0	▲	% Effort	72.0	%Schedule	60.0
24		Inception		▼	Inception		▼	Elaboration		▼	Elaboration		▼	Construction		Construction	
25																	
26	5.0	Distribute PM(PM_C) and M(M_BS) according to the Effort% & Schedule% per stage; and calculate P for the stage from its PM/M.															
36			Inception			Elaboration			Construction			Total E&C			Total		
37		Effort %	14.0			28.0			72.0			100.0			114.0		
38		Schedule%	40.0			40.0			60.0			100.0			140.0		
39		PrAve(P)	0.35			0.70			1.20			1.00			Does not apply		
40		PM/M=P	PM ×	M =	P	PM ×	M =	P	PM ×	M =	P	PM ×	M =	P	PM ×	M =	P-ave
41	BS	32000	19.91 ×	6.41 =	3.11	39.82 ×	6.41 =	6.22	102.39 ×	9.61 =	10.66	142.2 ×	16.0 =	8.9	162.1 ×	22.4 =	7.2
42																	
43	6.0	Get the Schedule Multipliers values.															
44						RVHL	N ▼	DPRS	VH ▼	CLAB	N ▼	RESL	N ▼	PPOS	N ▼		
45																	
46			Inception			Elaboration			Construction								
47			PM ×	M =	P	PM ×	M =	P	PM ×	M =	P						
48	N	RVHL	0.980	0.980	1.000	0.99	0.99	1.00	1.00	1.00	1.00						
49	VH	DPRS	0.900	0.900	1.000	0.950	0.950	1.000	0.950	0.950	1.000						
50	N	CLAB	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000						
51	N	RESL	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000						
52	N	PPOS	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000						
53		Π	0.882	0.882	1.000	0.941	0.941	1.000	0.950	0.950	1.000						
54																	

Figure 6. The COSSEMO extension and RAD Driver input portion of CORADMO.xls



COnstructive RAD schedule estimation MOdel (CoRADMo) Development Sheet: ED&PO-Dev

	A	B	C	E	G	H	J	L	M	O	Q	R	T	V	W	Y	AA
55	7.0	Apply the product of user selected Schedule Multipliers to each PM, M and P in each stage.															
56																	
57	8.0	Calculate PM and M "Total E&C" and "Total" (I-E-C) by adding the stages PM and M. Calculate P "Total E&C" and "Total" (I-E-C) by dividing total PM by total M															
60			Inception			Elaboration			Construction			Total E&C			Total		
61		Effort %	14.0			28.0			72.0			100.0			114.0		
62		Schedule %	40.0			40.0			60.0			100.0			140.0		
63		P/Ave(P)	0.35			0.70			1.20			1.00					
64		PM/M=P	PM	M	P	PM	M	P	PM	M	P	PM	M	P	PM	M	P-ave
65	BS	12000.00	19.91	6.41	3.11	39.82	6.41	6.22	102.09	9.61	10.68	142.2	16.0	8.9	162.1	22.4	7.2
66		n	0.88	0.88	1.00	0.94	0.94	1.00	0.95	0.95	1.00						
67	RAD	Eff&Schd	17.58	5.65	3.11	37.45	6.00	6.22	97.27	9.13	10.66	134.7	15.2	8.9	152.0	20.8	7.3
70																	
71	9.0	Courtesy plot of P vs M															

Figure 7. The RAD extension calculation and display of Schedule and Effort

Appendix 2, Part 2
CORADMO Drivers & Rating Scales

A. Winsor Brown
AWBrown@sunset.USC.edu

Outline

The five explicit drivers

- Reuse and Very High Level Languages (VHLL) (RVHL)
- Development Process Reengineering (DPRS)
- Collaboration Technology (CLAB)
- Architecture, Risk Resolution (RESL)
- Prepositioning Assets (PPOS)

Each is presented with

- Major factors influencing selection
- Statement of applicability to effort or schedule or both
- Rating levels to Numeric value conversion tables

Reuse and VHLLs (RVHL)

Standard 3GL module reuse: no adjustment

Schedule compression in Inception and Elaboration stages due to faster prototyping, option exploration

- effect depends on level of capability and experience in doing this (similar to Rapid Prototyping experience)
- same effect on effort; staff level held constant

Schedule and Effort Multipliers	Rapid Prototyping Experience Level				
	VL	L	N	H	VH
Inception	1.04	1.0	.98	.94	.90
Elaboration	1.02	1.0	.99	.97	.95
Construction	1.0	1.0	1.0	1.0	1.0

Development Process Reengineering and Streamlining (DPRS)

Detailed rating scale provided below

Gains depend on current level of bureaucracy

- Same effect on effort; staff level held constant

Schedule and Effort Multipliers	Inception	Elaboration	Construction
VL - Heavily Bureaucratic	1.20	1.15	1.15
L - Bureaucratic	1.08	1.06	1.06
N - Basic good business practices	1.0	1.0	1.0
H - Partly streamlined	.96	.98	.98
VH - Fully streamlined	.90	.95	.95

DPRS Rating Scale

	VL	L	N	H	VH
Number of approvals required per task	Excessive	Occasionally Reduced	Mature	Actively Reduced	Actively Minimized
Time taken per approval	Excessive	Occasionally Reduced	Mature	Actively Reduced	Actively Minimized
Reduced task dependencies, critical path tasks	None	Little	Mature Tech. Adopted	Advanced Tech. Adopted	Pioneering
Followup to expedite task completion	None	Little	Encouraged	Emphasized	Strongly Emphasized
Process measurement & streamlining	None	Little	Mature Tech. Adopted	Advanced Tech. Adopted	Pioneering

Collaboration Efficiency (CLAB)

Detailed rating scale provided

- Judgement-based average of COCOMO II ratings: SITE, TEAM & PREX
- SITE ratings also include
  - collaboration tool maturity, experience
  - scope effects: domain, negotiation, option-analysis tool support

Same effect on effort; staff level held constant

Schedule & Effort Multipliers	VL	L	N	H	VH	EH
Inception	1.21	1.10	1.00	0.93	0.86	0.80
Elaboration	1.15	1.07	1.00	0.95	0.90	0.86
Construction	1.10	1.05	1.00	0.98	0.95	0.93

CLAB Rating Scale

Judgement-based average of COCOMO II factors

- SITE
- TEAM
- PREX

	VL	L	N	H	VH	EH
SITE	<== COCOMO II Post-Arch. Ratings ==>				plus negotiation/tradeoff tools	
					basic	advanced
TEAM	<=== <=== <=== COCOMO II Scale Factor Ratings ===> ===> ===>					
PREX	(EL & VL) <=== <=== <=== COCOMO II Early Design Ratings ===> ===> ===>					

Architecture / Risk Resolution (RESL)

Same as COCOMO II RESL rating scale

Enables parallel construction

- Assumes higher level of staffing available and used (case b & c on next page)
- Otherwise no schedule compression (case a on next page)

Schedule Multipliers (Effort Unchanged)	VL	L	N	H	VH	EH
Inception	1.0	1.0	1.0	1.0	1.0	1.0
Elaboration	1.0	1.0	1.0	1.0	1.0	1.0
Construction	1.0	1.0	1.0	.91	.83	.75

Prepositioning Assets (PPOS)

Degree to which assets are pre-tailored to project and furnished to project for use on demand

- People skills and teambuilding
- Processes and tools
- Architecture and componentry

Requires product-line approach

or added (3, 6, 10%) pre-LCO (e.g. Inception) effort investment

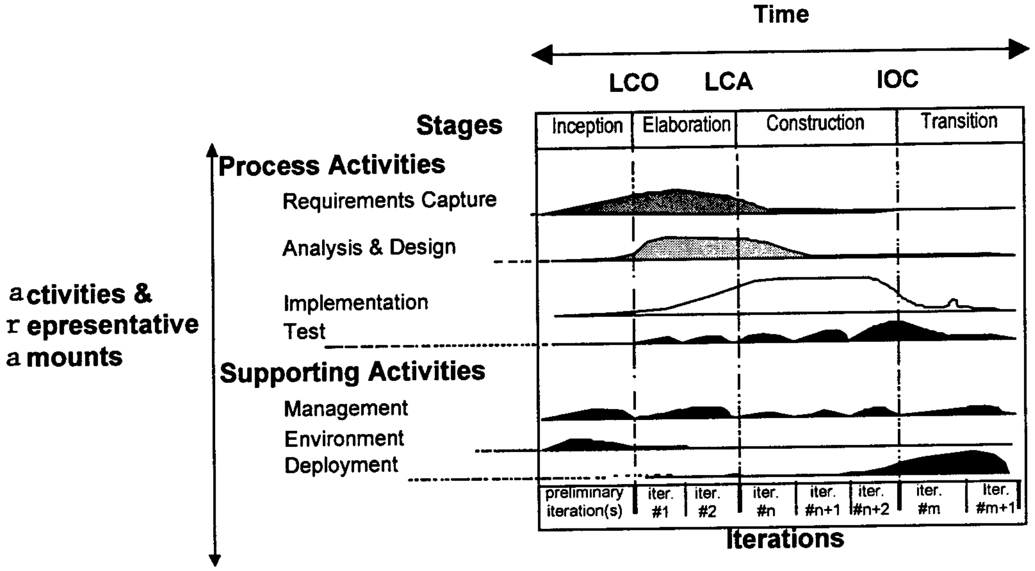
PM/M=P Multipliers	N	H	VH	EH
Rating	Basic project legacy, no tailoring	Some prepositioning & tailoring	Key items prepositioned & tailored	All items prepositioned & tailored
Inception	1.0/1.0=1.0	1.03/.93=1.11	1.06/.86=1.23	1.1/.80=1.37
Elaboration	1.0/1.0=1.0	1.03/.93=1.11	1.06/.86=1.23	1.1/.80=1.37
Construction	1.0/1.0=1.0	1.03/.93=1.11	1.06/.86=1.23	1.1/.80=1.37[1]

[1] Interpretation of (Construction,EH) table entry: PM=1.1 (effort multiplier; person months);
M=0.80 (schedule multiplier; months); P=1.37 (FSP multiplier; persons)

Appendix 2, part 3
CORADMO and COSSEMO Driver

1. COCOMO Stage Schedule and Effort MODEL (COSSEMO)

COSSEMO is based on the lifecycle anchoring concepts discussed by Boehm[3]. The anchor points are defined as Life Cycle Objectives (LCO), Life Cycle Architecture (LCA), and Initial Operational Capability (IOC). An enhanced version of an illustration from Rational Corporation[4] showing the stages around the anchor points is shown below.



The correspondence between COSSEMO's & CORADMO's "Stages"[5], COCOMOII's submodels and the life cycle anchor points is shown in the following table along with an indication of the relative amounts of the different activities.

[1] Constructive RAD schedule and effort Model

[2] COCOMO-II Staged Schedule and Effort Model

[3] Barry W. Boehm, "Anchoring the Software Process," *IEEE Software,* 13, 4, July 1996, pp. 73-82

[4] Rational Corp., "Rational Objectory Process 4.1 – Your UML Process", available at http://www.rational.com/support/techpapers/toratobjprcs/.

[5] COSSEMO & CORADMO use the word "stage" so it is not confused with the classic waterfall phases: Requirements, Analysis, Design, Code, Test and Maintenance.

CORADMO and COSSEMO Driver

COCOMO II Submodel Usage	Early Design		Post-Architecture	Maintenance
	LCO	LCA	IOC	
Activities \ Stage	**Inception**	**Elaboration**	**Construction**	**Transition**
Requirements Capture	Some usually	Most, peaks here	Minor	None
Analysis & Design	A little	Majority, mostly constant effort	Some	Some, for repair during ODT&E
Implementation	Practically none	Some, usually for risk reduction	Bulk; mostly constant effort	Some, for repair during ODT&E
Test	None	Some, for prototypes	Most for unit, integration and qualification test.	Some, for repaired code.

COCOMOII's effort and schedule estimates are focused on Elaboration and Construction (the stages between LCO and IOC. Inception corresponds to the COCOMO's "Requirements" activity in a waterfall process model. COCOMO's effort for the "Requirements" activity is an additional, fixed percentage of the effort calculated by COCOMO for the development activities. The table also indicates the areas in which the COCOMO II Early Design and Post-Architecture submodels are normally used.

CORADMO and COSSEMO Driver

Allocations

1.A.1. Percentage Effort per Stage. Allocate the effort (person months) used in each of the stages as a percentage of the total effort during Elaboration and Construction. The sum of the percentages of Elaboration and Construction should be 100%. The effort during Inception (as a percentage of total Elaboration and Construction) is added to get the Total IE&C which should be greater than 100%.

	LCO	LCA	IOC		
Stage	**Inception**	**Elaboration**	**Construction**	**Total E & C**	**Total I E & C**
%Effort				**100%**	

1.A.2. Percentage Schedule per Stage. Allocate the schedule (calendar months) for each of the stages as a percentage of the total schedule during Elaboration and Construction. The sum of Elaboration and Construction should be 100%. The schedule during Inception (as a percentage of total Elaboration and Construction) is added to get the Total IE&C which should be greater than 100%.

	LCO	LCA	IOC		
Stage	**Inception**	**Elaboration**	**Construction**	**Total E & C**	**Total I E & C**
%Schedule				**100%**	

1.A.3. Person-Power per Stage. Indicate the average number of people actually working during this period of each of the stages. If the loading was not approximately constant during the period except for typical, limited ramp-ups, please indicate the degree of variation by providing the Persons-Max and Persons-Min, and the number of months with that number of people (max and min, respectively). NOTE: summing persons across stages is illogical and incorrect.

CORADMO Driver Value Determination Worksheet

		LCO		LCA		IOC		
Stage	**Inception**		**Elaboration**		**Construction**		**Total E & C**	**Total I E & C**
Persons, Average							X	X
	Heads	Months	Heads	Months	Heads	Months	X	X
Persons, Maximum							X	X
Persons, Minimum							X	X

CORADMO Driver Value Determination Worksheet

2. COCOMO RAD MODEL (CORADMO)

The intent of the COCOMO II RAD model is to calculate/predict the schedule (months, M), personnel (P), and adjusted effort (person-months, PM) based on the distribution of effort and schedule to the various stages, and impacts of the selected schedule driver ratings on the M, P, and PM of each stage.

2.A.1. Reuse and VHLL's (RVHL) The degree to which re-use of other than code and/or very high level languages are utilized. This driver reflects schedule compression in Inception and Elaboration stages due to faster prototyping or option exploration. The rating for this driver depends on the amount of Rapid Prototyping Experience the development team has had in the domain of the project being evaluated. Since the rating applies to the team, it must include the experience of the managers and team leaders and their experience takes precedence over the average of the rest of the team working in the Inception and Elaboration phases.

RVHL		Very Low	Low	Nominal	High	Very High
Don't Know	N/A - Not Applicable	none	On average, personnel have experience on less than one recent project using Rapid Prototyping	most personnel have worked on more than one project using Rapid Prototyping	on average, personnel have worked on more than two projects using Rapid Prototyping	all personnel have worked on at least three projects using Rapid Prototyping

N/A
rationale:____
____

CORADMO Driver Value Determination Worksheet

2.A.2. Development Process Reengineering and Streamlining (DPRS) The degree to which the project and organization allow and encourage streamlined or re-engineered development processes: the current level of bureaucracy is a clear indicator. The schedule compression or expansion, because of this driver, doesn't alter staff level (P). The following table can be used to make a subjective average to determine the level of bureaucracy.

Level of Bureaucracy Indicators	Very Low	Low	Nominal	High	Very High
Number of approvals required per task	Excessive	Occasionally Reduced	Mature	Actively Reduced	Actively Minimized
Time taken per approval	Excessive	Occasionally Reduced	Mature	Actively Reduced	Actively Minimized
Reduced task dependencies, critical path tasks	None	Little	Mature Tech. Adopted	Advanced Tech. Adopted	Pioneering
Follow-up to expedite task completion	None	Little	Encouraged	Emphasized	Strongly Emphasized
Process measurement & streamlining	None	Little	Mature Tech. Adopted	Advanced Tech. Adopted	Pioneering
Level of Bureaucracy	Heavily Bureaucratic	Bureaucratic	Basic good business practices	Partly streamlined	Fully streamlined

DPRS		Very Low	Low	Nominal	High	Very High
Don't Know	N/A - Not Applicable	Heavily Bureaucratic	Bureaucratic	Basic good business practices	Partly streamlined	Fully streamlined

N/A
rationale:____

CORADMO Driver Value Determination Worksheet

2.A.3. Collaboration Efficiency (CLAB) Teams and team members who can collaborate effectively can reduce both effort and schedule; those that don't collaborate effectively have increased schedule and effort (due to wasted time). With this multiplier, staff level does not change based on collaboration efficiency.

Collaboration efficiency is clearly impacted by TEAM and SITE ratings. Collaboration efficiency is impacted by TOOL, but only for tools that support or enable collaboration. However, the tool technology impact is lessened in the case of a co-located team with high experience ratings (PREX, the combination of application, platform, language and tool experience taken from the early design ratings).

2.A.3.1. Team Cohesion (TEAM). The Team Cohesion cost driver accounts for the sources of project turbulence and extra effort due to difficulties in synchronizing the project's stakeholders: users, customers, developers, maintainers, interfacers, others. See the Model Definition Manual for more details.

Team		Very Low	Low	Nominal	High	Very High	Extra High
Don't Know	**N/A - Not Applicable**	Very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions

N/A
rationale:____

CORADMO Driver Value Determination Worksheet

2.A.3.2. Multisite Development (SITE). Given the increasing frequency of multisite developments, and indications that multisite development effects are significant, the SITE cost driver has been added in COCOMO II. Determining its cost driver rating involves the assessment and averaging of two factors: site collocation (from fully collocated to international distribution) and communication support (from surface mail and some phone access to full interactive multimedia). See the COCOMO-II User's Manual. We recommend a 70% and 30% weightings for Collocation and Communications, respectively, when making your subjective average of these two components of SITE.

	Very Low	Low	Nominal	High	Very High	Extra High
SITE: Collocation	Inter-national	Multi-city and Multi-company	Multi-city or Multi-company	Same city or metro area	Same building or complex	Fully collocated
SITE: Communications	Some phone, mail	Individual phone, FAX	Narrowband email	Wideband electronic communication	Wideband elect. comm., occasional video conf.	Interactive multimedia

Don't Know	N/A	Very Low	Low	Nominal	High	Very High	Extra High

N/A
rationale:____

2.A.3.3. Applications Experience (AEXP). This rating is dependent on the level of applications experience of the project team developing the software system or subsystem. The ratings are defined in terms of the project team's equivalent level of experience with this type of application. See the COCOMO-II User's Manual.

AEXP		Very Low	Low	Nominal	High	Very High
Don't Know	N/A - Not Applicable	≤ 2 months	6 months	1 year	3 years	≥6 years

N/A
rationale:____

CORADMO Driver Value Determination Worksheet

2.A.3.4. Platform Experience (PEXP). The Post-Architecture model broadens the productivity influence of PEXP, recognizing the importance of understanding the use of more powerful platforms, including more graphic user interface, database, networking, and distributed middleware capabilities. See the COCOMO-II User's Manual.

PEXP		Very Low	Low	Nominal	High	Very High
Don't Know	N/A - Not Applicable	≤ 2 months	6 months	1 year	3 years	≥6 years

N/A
rationale:____

2.A.3.5. Language and Tool Experience (LTEX). This is a measure of the level of programming language and software tool experience of the project team developing the software system or subsystem. See the COCOMO-II User's Manual.

LTEX		Very Low	Low	Nominal	High	Very High
Don't Know	N/A - Not Applicable	≤ 2 months	6 months	1 year	3 years	≥6 years

N/A
rationale:____

2.A.3.6. Personnel Experience (PREX) This Early Design cost driver combines the three Post-Architecture cost drivers application experience (AEXP), platform experience (PEXP), and language and tool experience (LTEX). While these three Post-Architecture ratings normally apply to a module, for CoRADMo they are applied across the entire project. Their individual rating information is given above.

The approach for mapping the Post-Architecture cost drivers and rating scales onto their Early Design model counterparts involves the use and combination of numerical equivalents of the rating levels. Specifically, a Very Low Post-Architecture cost driver rating corresponds to a numerical rating of 1, Low is 2, Nominal is 3, High is 4, Very High is 5, and Extra High is 6. For the combined Early Design cost drivers, the numerical values of the contributing Post-Architecture cost drivers are summed, and the resulting totals are allocated to an expanded Early Design model rating scale going from Extra Low to Extra High. The

CORADMO Driver Value Determination Worksheet

Early Design model rating scales always have a Nominal total equal to the sum of the Nominal ratings of its contributing Post-Architecture elements.

The table below assigns PREX ratings across this range, and associates appropriate effort multipliers and rating scales to each of the rating levels.

PREX		Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Sum of AEXP, PEXP, and LTEX ratings		3, 4	5, 6	7, 8	9	10, 11	12, 13	14, 15
Applications, Platform, Language and Tool Experience		≤ 3 mo.	5 months	9 months	1 year	2 years	4 years	≥ 6 years
Don't Know	**N/A - Not Applicable**							

CORADMO Driver Value Determination Worksheet

To determine the CLAB rating, take the subjective/fuzzy average of TEAM and SITE ratings from COCOMO II's post-architecture definitions and the PREX ratings using COCOMO II's Early Design definitions.

	Very Low	Low	Nominal	High	Very High	Extra High
SITE	<== COCOMO II Post-Arch. Ratings ==>				High plus negotiation/tradeoff tools basic	advanced
TEAM	←= ←= ←= COCOMO II Scale Factor Ratings =→ =→ =→					
PREX	(EL & VL) <=== <=== <=== COCOMO II Early Design Ratings ===> ===> ===>					
Fuzzy Average						

CLAB		Very Low	Low	Nominal	High	Very High	Extra High
Don't Know	N/A - Not Applicable	← Pick most appropriate rating level based on fuzzy average →					

N/A
rationale:____

CORADMO Driver Value Determination Worksheet

2.A.4. Architecture & Risk Resolution (RESL) This rating is exactly the same as the COCOMO II RESL rating. The architecture portion enables parallel construction, thus reducing schedule during the construction phase assuming that staff level increases during construction while applying the same effort. Good risk resolution in a schedule driven development effort applying RAD strategies increases the probability of the strategies success.

RESL		Very Low	Low	Nominal	High	Very High	Extra High
Don't Know	**N/A - Not Applicable**	<== Use COCOMO II's RESL Rating Level ==>					

N/A
rationale:____

2.A.5. Prepositioning Assets (PPOS) This driver assesses the degree to which assets are pre-tailored to a project and furnished to the project for use on demand. This clearly has impacts from people skills and team building. The assets that are being pre-positioned include processes and tools, and architecture and componentry.

In order to take advantage of PPOS, the organization must either be taking a product-line approach or have made a 3, 6 or 10% pre-Inception effort investment!

PPOS		Nominal	High	Very High	Extra High
Don't Know	**N/A - Not Applicable**	Basic project legacy, no tailoring	Some prepositioning and tailoring	Key items prepositioned and tailored	All items prepositioned and tailored

N/A
rationale:____

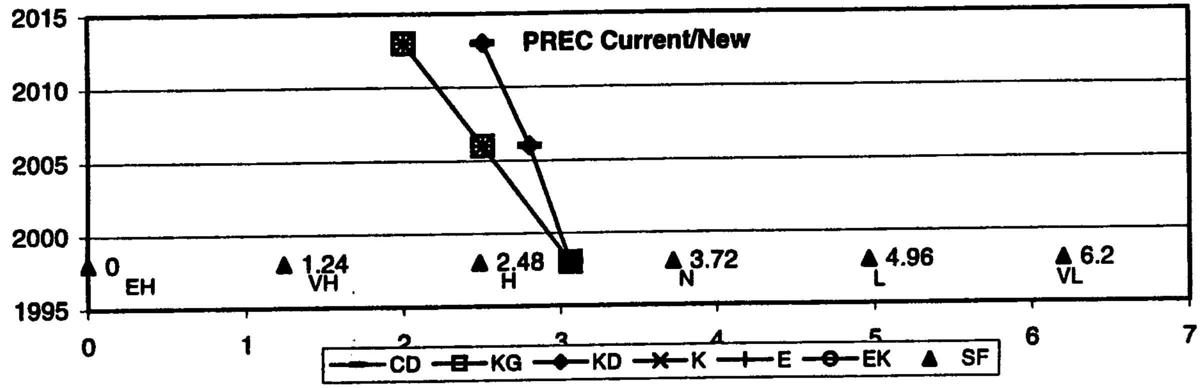
M. Appendix 3

Impact Analysis Worksheets and Calculations

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

PREC: Precedentedness

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
PREC default	3.06	2.80	2.50	2.50	2.00	2.80	2.50	2.50	2.00	2.50	2.00	2.50	2.00
PREC new													



CD: Some gains due to better general understanding of EHART domains, but not large

KG: Solid gains due to stronger domain understanding and technology, but offset by continuing need for more advanced systems

KD: No additional domain gains over CD

K: Same as KG

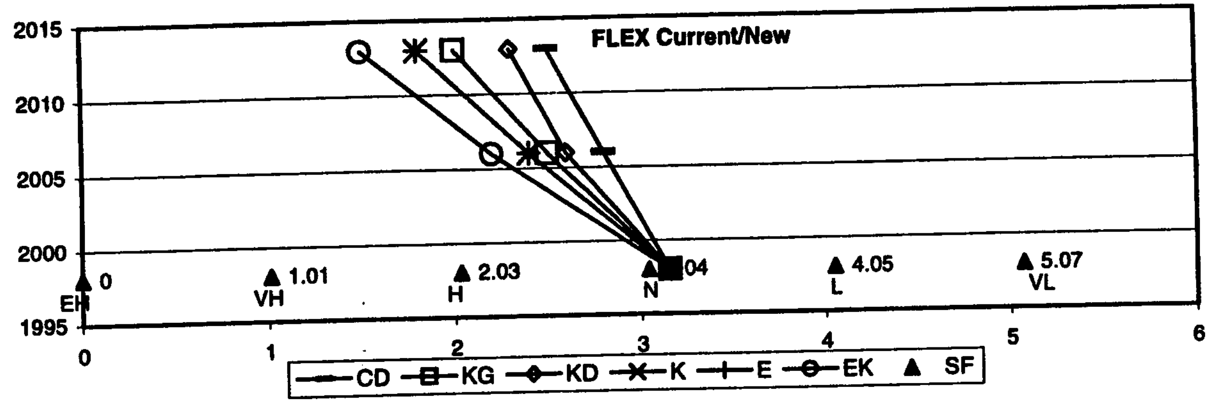
E: No additional domain gains over KG

EK: Same as E and K

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

FLEX: Development Flexibility

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
FLEX default	3.15	2.80	2.50	2.50	2.00	2.60	2.30	2.40	1.80	2.40	1.80	2.20	1.50
FLEX new													



CD: Some gains due to reduced project constraints via more flexible DoD acquisition, open systems and productivity gains

KG: Some similar added gains over CD, mainly via domain understanding

KD: Some similar added gains over CD, mainly via project understanding

K: Some complementary gains from KG and KD

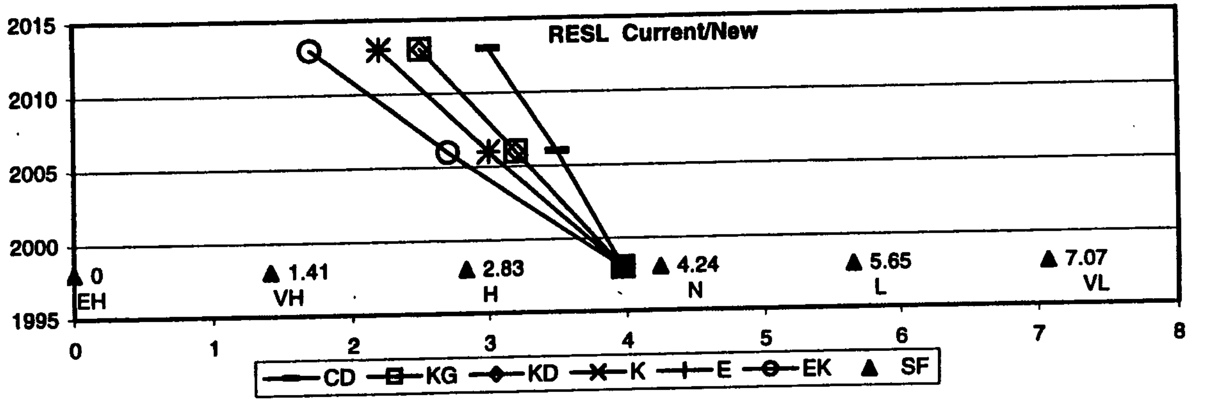
E: Some similar added gains over CD, mainly via high assurance technology and dynamic languages

EK: Some complementary gains from E and K

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

RESL: Architecture/Risk Resolution

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
RESL default	3.97	3.50	3.00	3.20	2.50	3.20	2.50	3.00	2.20	3.00	2.20	2.70	1.70
RESL new													



CD: Significant gains due to open systems technology, commercial OO architecture technology and DoD emphasis on risk management

KG: Significant additional gains over CD via domain architectures

KD: Significant additional gains over CD via architecture and risk advisor technology

K: Complementary gains from KG and KD

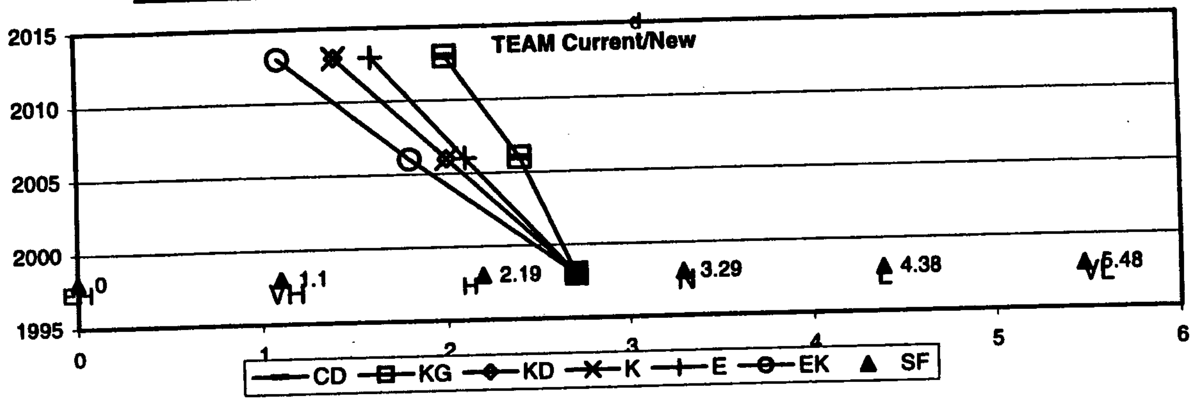
E: Major additional gains over CD via domain architecture, general architecture technology, high assurance technology, and rationale capture

EK: Complementary gains from E and K

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

TEAM: Team Cohesion

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
TEAM default	2.70	2.40	2.00	2.40	2.00	2.00	1.40	2.00	1.40	2.10	1.60	1.80	1.10
TEAM new													



CD: Significant gains due to commercial collaboration technology, DoD emphasis on Integrated Product Teams (IPTs)

KG: No gains over CD

KD: Significant additional gains over CD due to KB collaboration technology, particularly in longer term

K: Same as KD

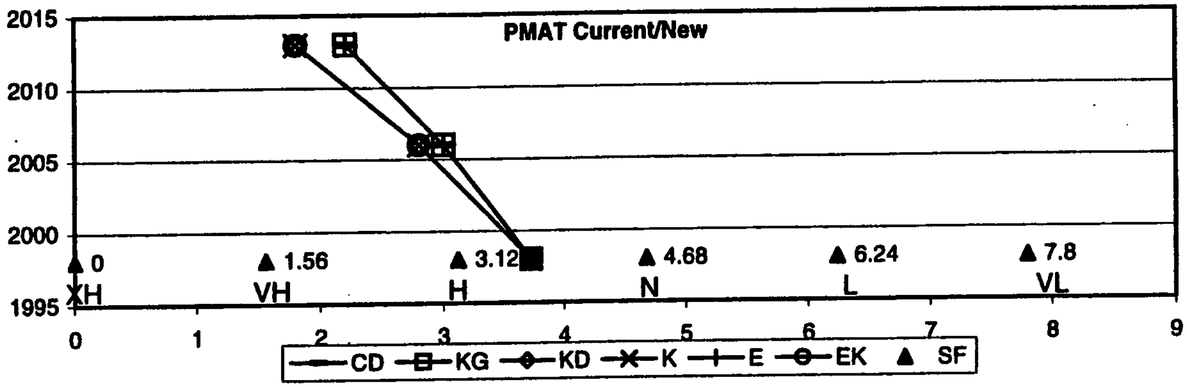
E: Significant additional gains over CD due to advanced collaboration technology

EK: Some complementary gains from E and K

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

PMAT: Process Maturity

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
PMAT default	3.72	3.00	2.20	3.00	2.20	2.80	1.80	2.80	1.80	3.00	2.20	2.80	1.80
PMAT new													



CD: Continuing major gains in process maturity due to continuing DoD emphasis and improving commercial tool support

KG: No gains over CD

KD: Moderate gains over CD via KB process technology

K: Same as KD

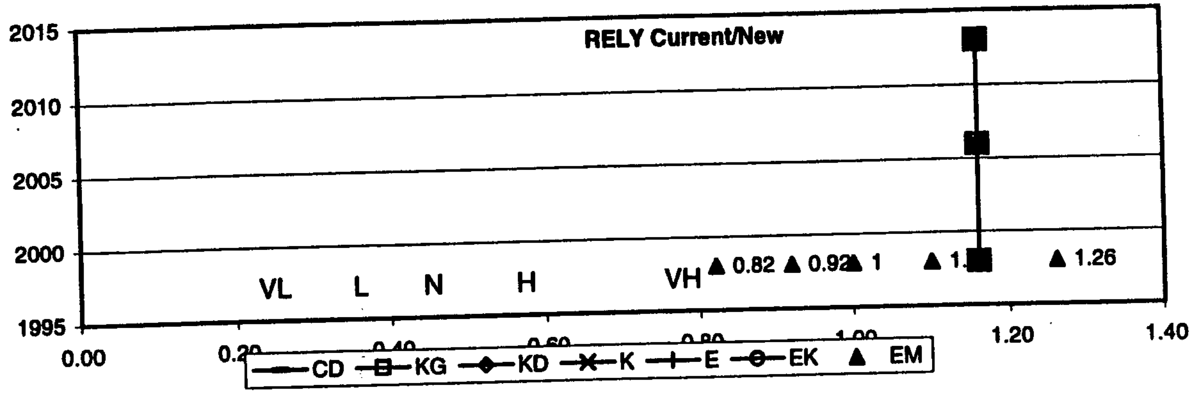
E: No gains over CD

EK: Same as K

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

RELY: Required Reliability

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
RELY default	1.16	1.14	1.12	1.14	1.12	1.14	1.12	1.14	1.12	1.10	1.06	1.10	1.06
RELY new													



Baselin Increased from 1.06 (COCOMO II sample mean) to 1.16 to reflect high-assurance EHART domain specifics

CD: Some gains due to commercial high-assurance technology reducing effort to achieve high assurance

KG: Same as CD

KD: Same as CD

K: Same as CD

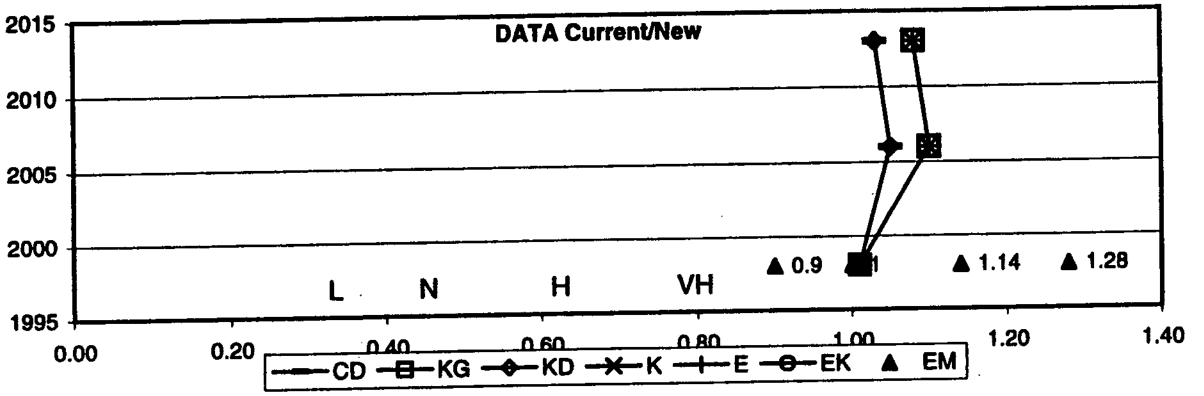
E: Significant gains over CD due to EDCS high-assurance technology

EK: Same as E

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

DATA: Data Amounts

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
DATA default	1.04	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98
DATA new													



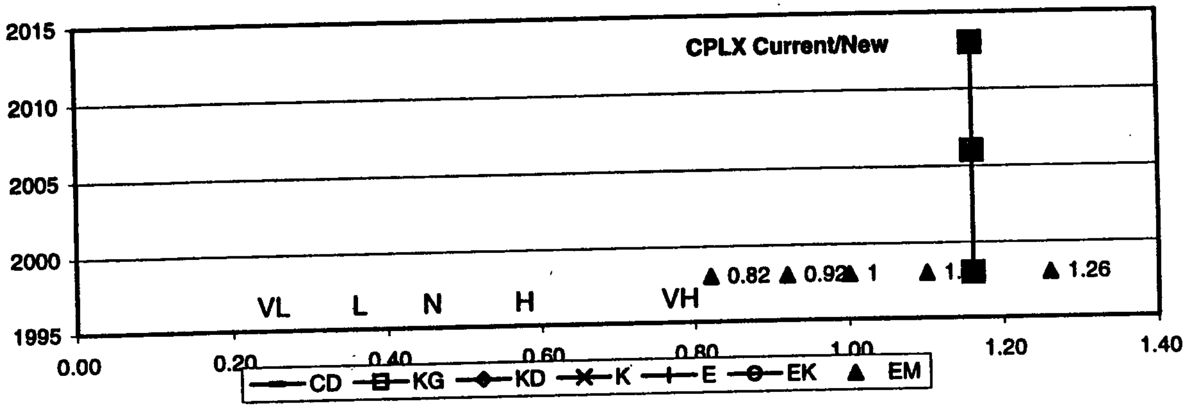
CD: Some gains due to improved commercial database migration, construction, and integrity maintenance capabilities

KG, KD, K, E, EK: Same as CD. No major database element in KBSA or EDCS programs.

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

CPLX: Product Complexity

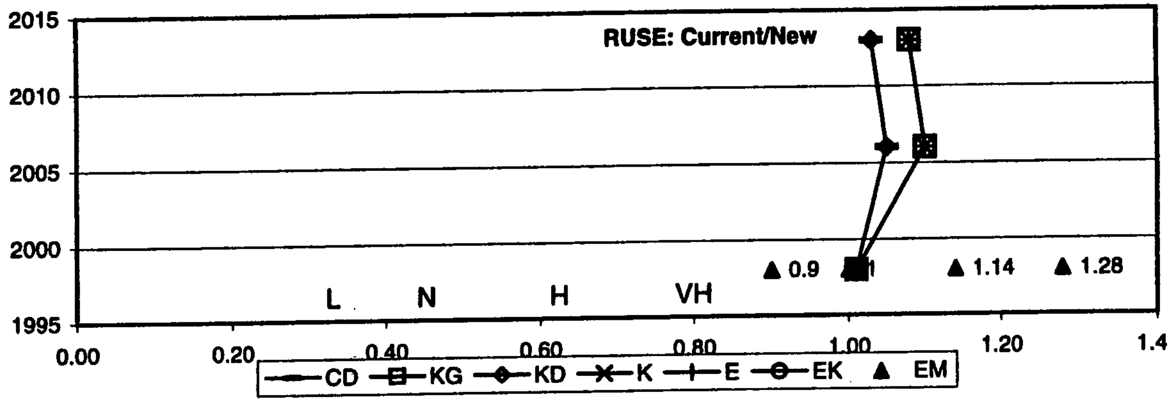
Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
CPLX default	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16
CPLX new													-



CD,
KG, EHART domain applications will remain complex. Domain understanding gains will largely be
KD, balanced by intoduction of more complex algorithms and architectures to increase system
K, E, accuracy, survivability, or other attributes
EK:

RUSE: Development for Reuse

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
RUSE default	1.01	1.05	1.03	1.10	1.08	1.05	1.03	1.10	1.08	1.10	1.08	1.10	1.08
RUSE new													-



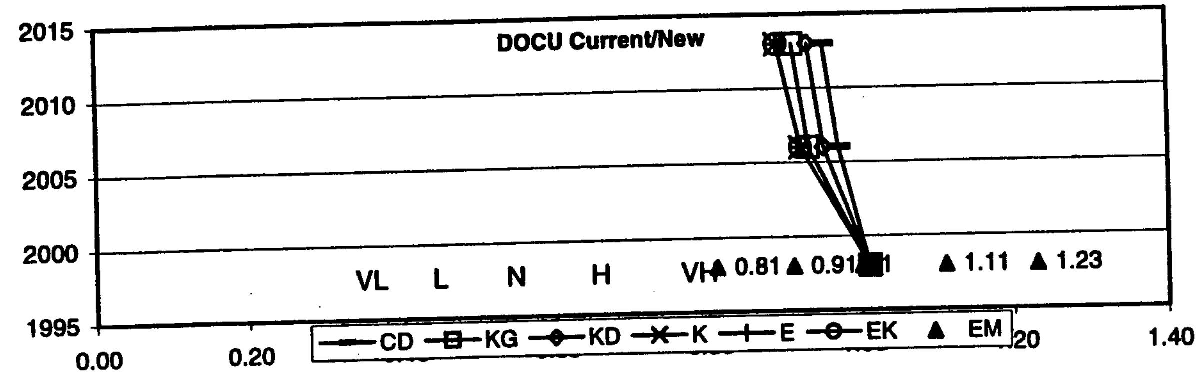
CD, KD: Moderate reuse gains (realized as reductions in the SIZE parameter) will require some added investment. This will be offset in the longer term by more efficient reuse technology

KG, K, E, EK: Significant added reuse gains over CD (via SIZE raductions) will require more investment, but again will be offset by more efficient domain and reuse technology

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

DOCU: Documentation Match to Needs

Driver	Baseline	CD@+8	CD@+15	KG@+8	KG@+15	KD@+8	KD@+15	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
DOCU default	1.01	0.97	0.95	0.93	0.91	0.95	0.93	0.92	0.89	0.92	0.89	0.92	0.89
DOCU new													



CD: Significant gains via move to DoD/commercial standards such as IEEE/EIA 12207.1 specifying use of "data" vs. "documentation," general commercial documentation aids

KG: Significant additional gains over CD via EHART domain-specific documentation/understanding aids

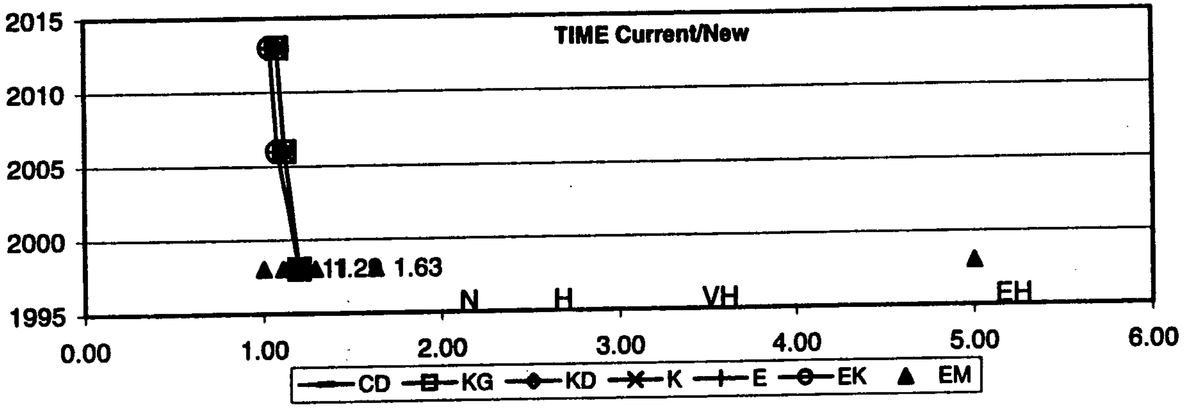
KD: Some other gains over CD via KB documentation/understanding aids

K, E, EK: Full KBSA technology very similar to full EDCS technology in this area

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

TIME: Execution Time Constraint

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
TIME default	1.20	1.12	1.08	1.12	1.08	1.12	1.08	1.12	1.08	1.08	1.04	1.08	1.04
TIME new													



Baselin(Increased from 1.08 to 1.20 to reflect real-time EHART domain specifics

CD: Major gains via Moore's Law hardware speedups, partly offset by use of extra cycles to improve accuracy, survivability, etc.

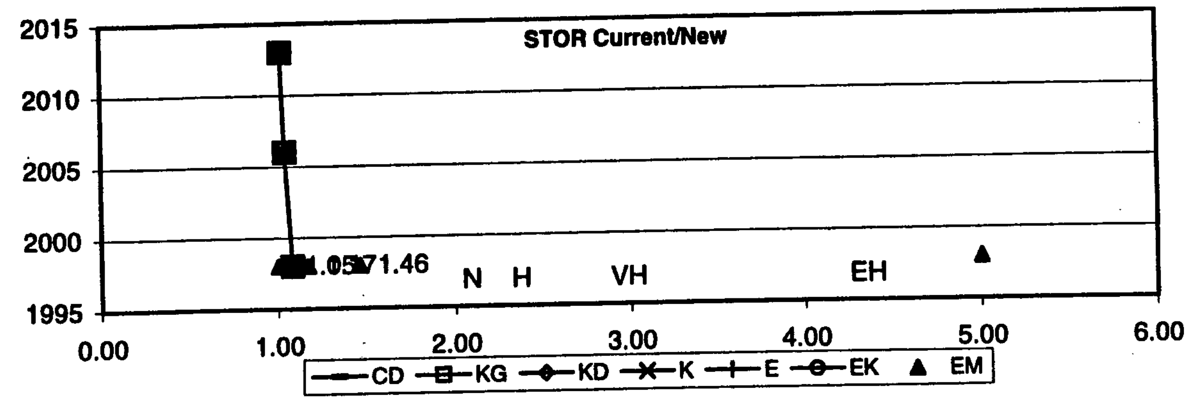
KG, KD, K: Same as CD

E, EK: Significant additional gains over CD via EDCS real-time and test/verification technology

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

STOR: Main Storage Constraint

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
STOR default	1.08	1.04	1.02	1.04	1.02	1.04	1.02	1.04	1.02	1.04	1.02	1.04	1.02
STOR new													



Baselin(Increased from 1.03 to 1.08 to reflect EHART embedded domain specifics

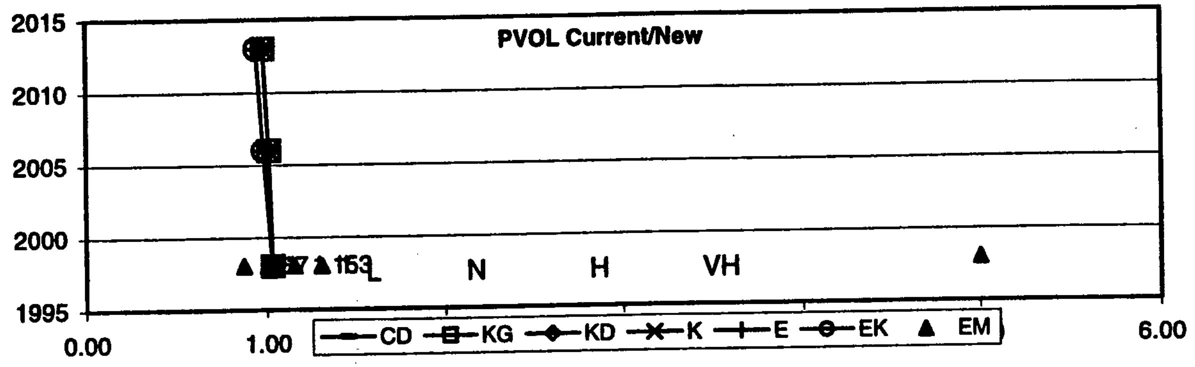
CD: Major gains via Moore's Law hardware capacity growth, partly offset by use of extea memory to improve accuracy, survivability, etc.

KG, KD, K, E, EK: Same as CD

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

PVOL: Platform Volatility

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
PVOL default	1.03	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	0.98	0.94	0.98	0.94
PVOL new													



CD: Pace of technology (particularly distributed processing) will keep platforms volatile, but the field will mature somewhat

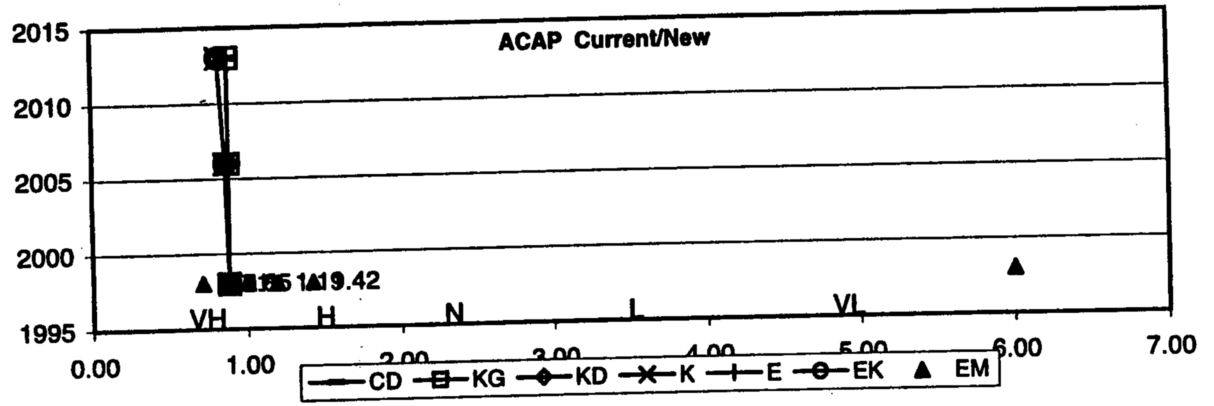
KG, KD, K: Same as CD

E, EK: EDCS EHART domain architectures will reduce platform volatility effects significantly

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

ACAP: Analyst Capability

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
ACAP default	0.88	0.88	0.88	0.88	0.88	0.86	0.82	0.86	0.82	0.88	0.88	0.86	0.82
ACAP new													



CD: No net change in personnel capability available for EHART application

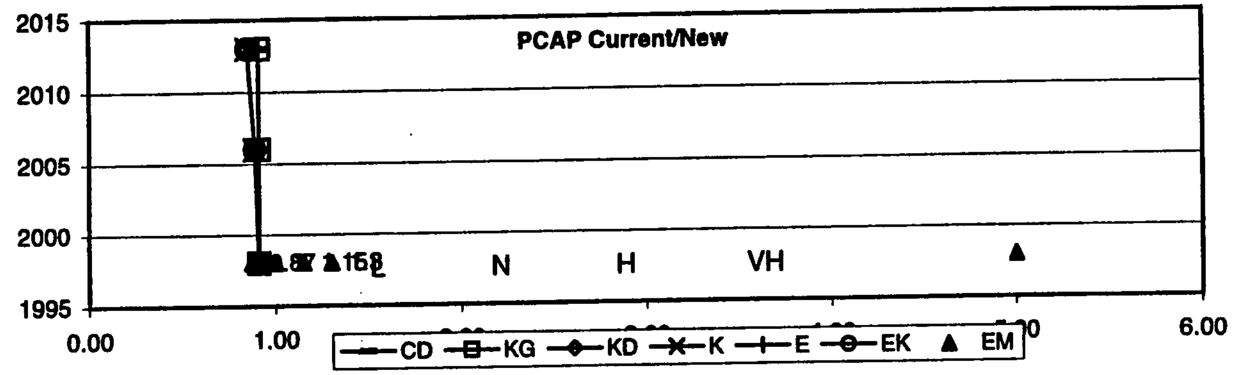
KG, E: Same as CD

KD, K, EK: Significant gains over CD in effective analyst capability via KB software decision aids

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

PCAP: Programmer Capability

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
PCAP default	0.91	0.91	0.91	0.91	0.91	0.89	0.85	0.89	0.85	0.91	0.91	0.89	0.85
PCAP new													



CD: No net change in programmer capability available for EHART applications

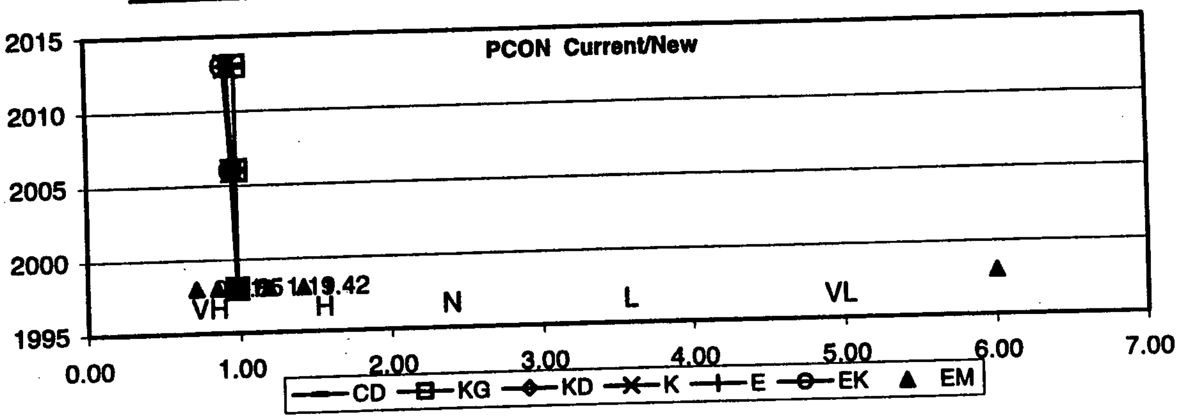
KG, E: Same as CD

KD, K, EK: Significant gains over CD in effective programmer capability via KB software decision aids

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

PCON: Personnel Continuity

Driver	Baseline	CD@+8	CD@+15	KG@+8	KG@+15	KD@+8	KD@+15	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
PCON default	0.98	0.98	0.98	0.98	0.98	0.96	0.93	0.96	0.93	0.96	0.93	0.95	0.91
PCON new													



CD: No net change in personnel continuity for EHART applications

KG: Same as CD

KD, K: Significant gains over CD in compensating for personnel turnover due to KB decision aids

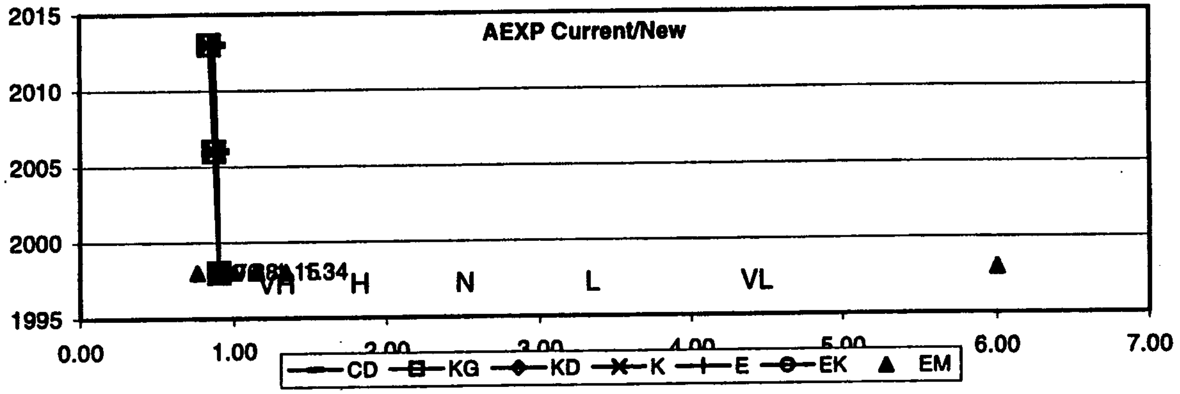
E: Significant gains over CD due to EDCS rationale capture and software understanding aids

EK: Complementary gains from E and K

AEXP: Application Experience

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
AEXP default	0.90	0.89	0.87	0.87	0.84	0.89	0.87	0.87	0.84	0.87	0.84	0.87	0.84
AEXP new													



CD: Some gains due to better general understanding of EHART domains, but not large

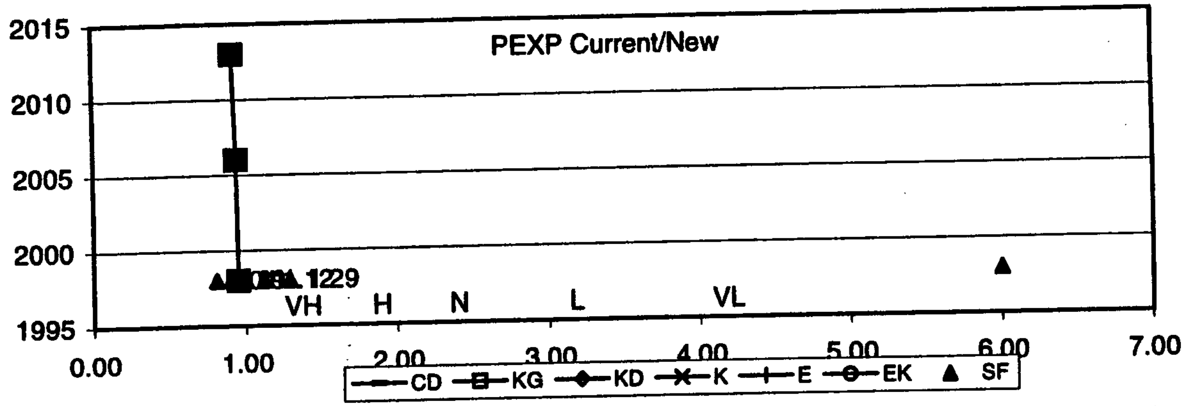
KD: Same as CD

KG,
K, E, Solid gains over CD due to stronger domain understanding and technology
EK:

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

PEXP: Platform Experience

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
PEXP default	0.95	0.94	0.92	0.94	0.92	0.94	0.92	0.94	0.92	0.94	0.92	0.94	0.92
PEXP new													



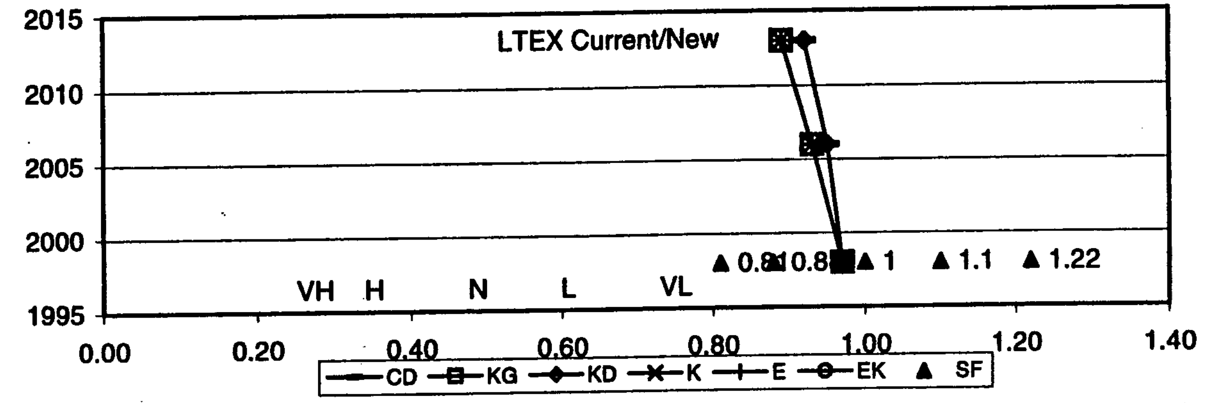
CD: Some gains due to platform maturity, but pace of technology (particularly distributed processing) will keep platform volatile and offset PEXP gains

KG,
KD,
K, E,
EK: Same as CD

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

LTEX: Language and Tool Experience

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
LTEX default	0.97	0.95	0.92	0.93	0.89	0.95	0.92	0.93	0.89	0.93	0.89	0.93	0.89
LTEX new													



CD: Some gains due to language and tool maturity and user aids; pace of technology will keep tools volatile and offset experience gains

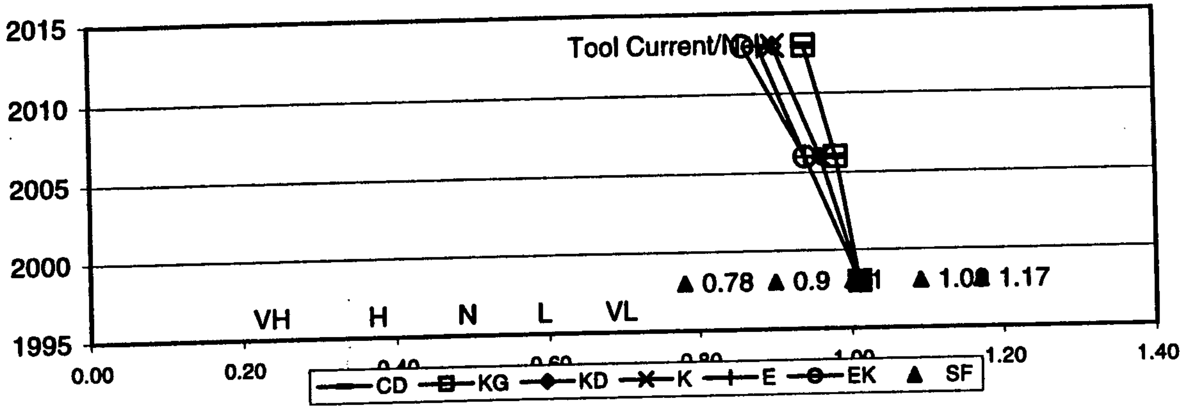
KD: Same as CD

KG,
K, E, EHART domain-specific languages and tools will provide solid gains over CD
EK:

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

TOOL: Use of Software Tools

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
TOOL default	1.01	0.98	0.94	0.98	0.94	0.96	0.90	0.96	0.90	0.94	0.88	0.94	0.86
TOOL new													



CD: Significant gains due to general tool technology capabilities

KG: Same as CD

KD, K: Solid gains over CD via KB decision support tools, particularly in longer term

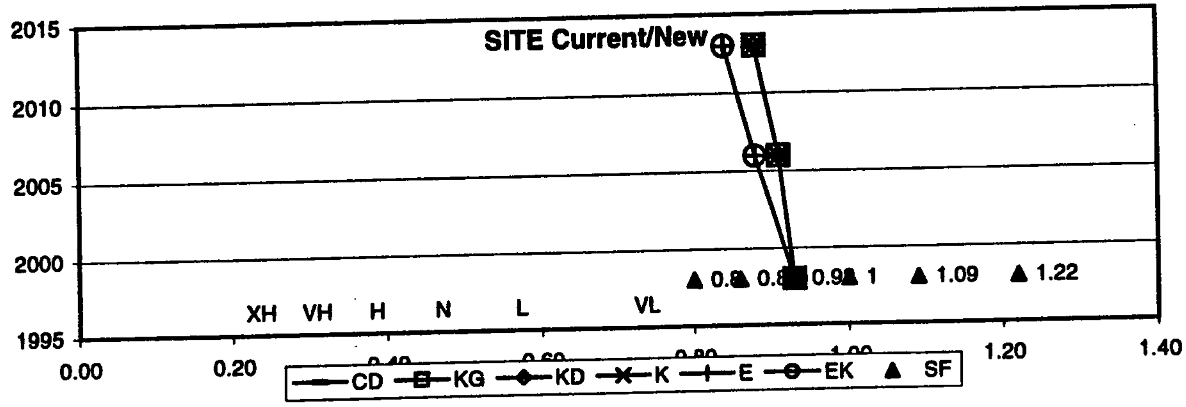
E: Significant gains over CD due to advanced EDCS tool capabilities, particularly in evolution support and EHART domain tools

EK: Complementary combination.

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

SITE: Multisite Development

Driver		Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
SITE	default	0.93	0.91	0.88	0.91	0.88	0.91	0.88	0.91	0.88	0.88	0.84	0.88	0.84
SITE	new													



CD: Improvements in distributed support offset by increased project distribution

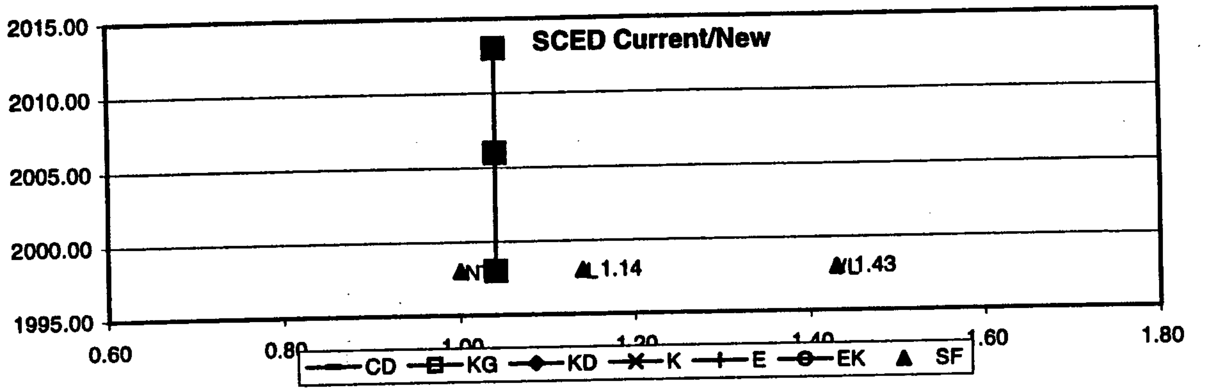
KG, KD, K: Same as CD

E, EK: Solid gains over CD due to EDCS distributed collaboration technology

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

SCED: Required Development Schedule

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
SCED default	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04
SCED new													

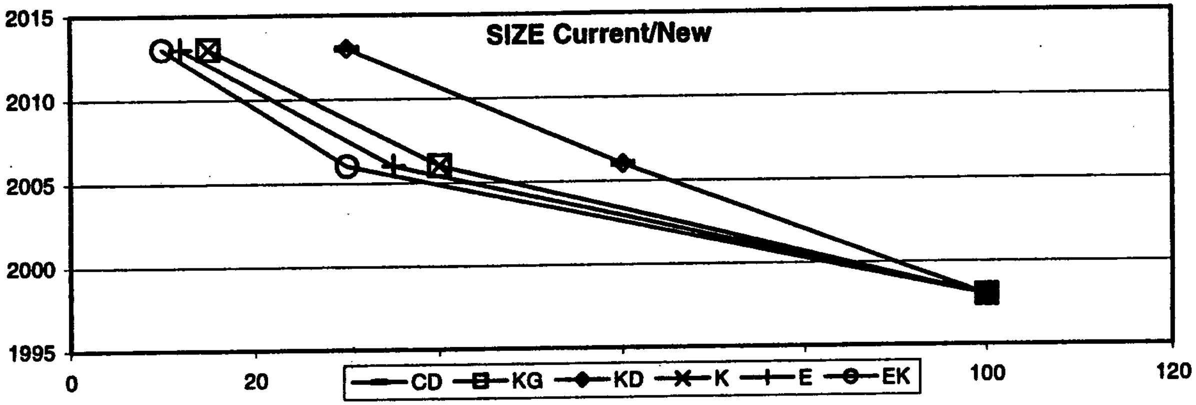


ALL: Schedule compression effects being covered by Rapid Application Development (RAD) model

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

SIZE: KSLOC

Driver		Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
SIZE	default	100.0	60.0	30.0	40.0	15.0	60.0	30.0	40.0	15.0	35.0	12.0	30.0	10.0
SIZE	new													



Size (KSLOC) is the primary determinant of software effort in COCOMO II (and other SCE models).
For COCOMO II, effective size = f(KSLOC or FP, BRAK, ADSI, DM, CM, IM, SU, UNFM).

Baseline is a 100 KSLOC embedded, high-assurance, real-time (EHART) software application.

CD: Commercial technology will provide better reuse infrastructure (e.g. IRL's??) and some of the componentry technology need for EHART applications. Better requirements technology will reduce breakage somewhat. The overall effects for EHART applications wi

KD: Same as CD

KG, K Significant gains over CD due to EHART domain-specific architectures, reuse, and application generators

E: Similar domain-specific gains, plus additional reduced breakage due to requirements and rationale capture technology, and reduced software understanding penalties due to software understanding technology

EK: Gains over E due to stronger KB application generator technology

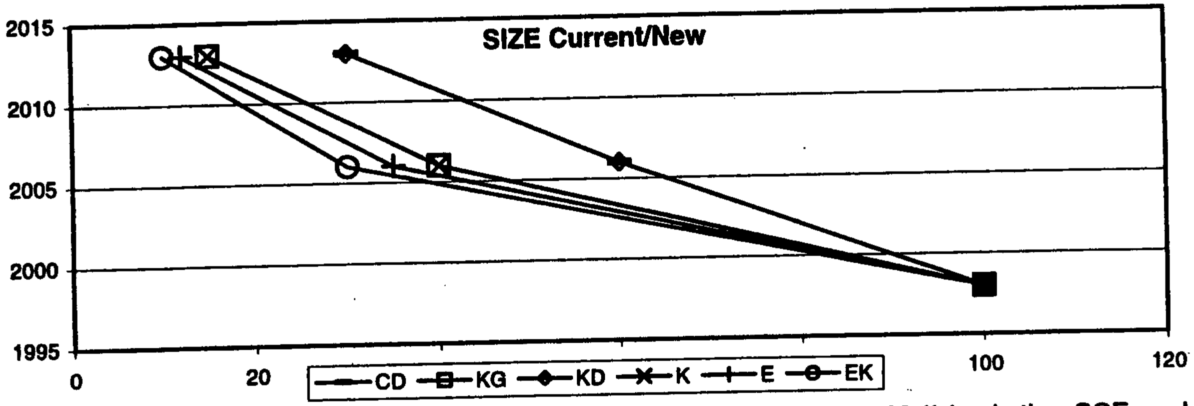
Results are conservative, particularly for EDCS, as maintenance savings would be greater than development savings, due to reductions in amount of software understanding, redesign, recode, and retest effort.

App-3-1--CII Drivers

Technology Impact Analyzer -- Individual CoCoMoII Scale Factors & Effort Multiplier values with Rationales over time: 1998, 2006 & 2013

SIZE: KSLOC (repeated for new values and justifcation)

Driver		Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
SIZE	default	100.0	60.0	30.0	40.0	15.0	60.0	30.0	40.0	15.0	35.0	12.0	30.0	10.0
SIZE	new													



Size (KSLOC) is the primary determinant of software effort in COCOMO II (and other SCE models).
For COCOMO II, effective size = f(KSLOC or FP, BRAK, ADSI, DM, CM, IM, SU, UNFM).

Baseline is a 100 KSLOC embedded, high-assurance, real-time (EHART) software application.

Tech Impact Analyzer: COCOMOII-1998 Scale Factors & Effort Multipliers -- Now, +8 & +15 years
COCOMO II data (from COCOMO II Drivers) and calculations

Factors into future

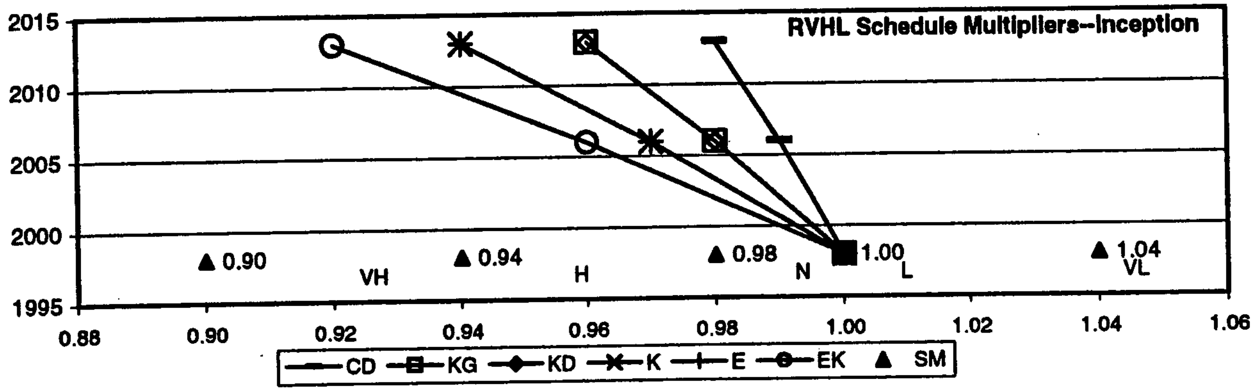
	Baseline	CD	CD	KG	KG	KD	KD	K	K	EDCS	EDCS	EK	EK	
Cost-	(now)	8Yr	15Yr	8Yr	15Yr	8Yr	15Yr	8Yr	15Yr	8Yr	15Yr	8Yr	15Yr	
Driver	1998	2006	2013	2006	2013	2006	2013	2006	2013	2006	2013	2006	2013	
PREC	3.1	2.8	2.5	2.5	2.0	2.8	2.5	2.5	2.0	2.5	2.0	2.5	2.0	
FLEX	3.2	2.8	2.5	2.5	2.0	2.6	2.3	2.4	1.8	2.4	1.8	2.2	1.5	
RESL	4.0	3.5	3.0	3.2	2.5	3.2	2.5	3.0	2.2	3.0	2.2	2.7	1.7	
TEAM	2.7	2.4	2.0	2.4	2.0	2.0	1.4	2.0	1.4	2.1	1.6	1.8	1.1	
PMAT	3.7	3.0	2.2	3.0	2.2	2.8	1.8	2.8	1.8	3.0	2.2	2.8	1.8	
Σ	16.6	14.5	12.2	13.6	10.7	13.4	10.5	12.7	9.2	13.0	9.8	12.0	8.1	
B	1.076	1.055	1.032	1.046	1.017	1.044	1.015	1.037	1.002	1.040	1.008	1.030	0.991	=0.91+0.01*Sigma
RELY	1.16	1.14	1.12	1.14	1.12	1.14	1.12	1.14	1.12	1.10	1.06	1.10	1.06	
DATA	1.04	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	
CPLX	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	
RUSE	1.01	1.05	1.03	1.10	1.08	1.05	1.03	1.10	1.08	1.10	1.08	1.10	1.08	
DOCU	1.01	0.97	0.95	0.93	0.91	0.95	0.93	0.92	0.89	0.92	0.89	0.92	0.89	
TIME	1.20	1.12	1.08	1.12	1.08	1.12	1.08	1.12	1.08	1.08	1.04	1.08	1.04	
STOR	1.08	1.04	1.02	1.04	1.02	1.04	1.02	1.04	1.02	1.04	1.02	1.04	1.02	
PVOL	1.03	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	0.98	0.94	0.98	0.94	
ACAP	0.88	0.88	0.88	0.88	0.88	0.86	0.82	0.86	0.82	0.88	0.88	0.86	0.82	
PCAP	0.91	0.91	0.91	0.91	0.91	0.89	0.85	0.89	0.85	0.91	0.91	0.89	0.85	
PCON	0.98	0.98	0.98	0.98	0.98	0.96	0.93	0.96	0.93	0.96	0.93	0.95	0.91	
AEXP	0.90	0.89	0.87	0.87	0.84	0.89	0.87	0.87	0.84	0.87	0.84	0.87	0.84	
PEXP	0.95	0.94	0.92	0.94	0.92	0.94	0.92	0.94	0.92	0.94	0.92	0.94	0.92	
LTEX	0.97	0.95	0.92	0.93	0.89	0.95	0.92	0.93	0.89	0.93	0.89	0.93	0.89	
TOOL	1.01	0.98	0.94	0.98	0.94	0.96	0.90	0.96	0.90	0.94	0.88	0.94	0.86	
SITE	0.93	0.91	0.88	0.91	0.88	0.91	0.88	0.91	0.88	0.88	0.84	0.88	0.84	
SCED	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	
Π	1.21	0.93	0.67	0.89	0.63	0.83	0.52	0.81	0.49	0.72	0.45	0.68	0.38	
ΠnoSCED	1.17	0.89	0.64	0.86	0.60	0.80	0.50	0.78	0.47	0.69	0.44	0.66	0.36	
SCED%	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	via linear interpolation
SIZE Orig.	100	60	30	40	.15	60	30	40	15	35	12	30	10	
SIZE	100	60	30	40	15	60	30	40	15	35	12	30	10	
CII_PM	505.48	204.55	65.76	124.00	28.97	175.65	48.05	108.83	21.51	85.66	16.37	66.71	10.90	
CII_PMnoSCED	486.04	196.69	63.23	119.23	27.85	168.90	46.20	104.65	20.68	82.37	15.74	64.15	10.48	
CII_PM Orig.	505.48	204.55	65.76	124.00	28.97	175.65	48.05	108.83	21.51	85.66	16.37	66.71	10.90	
CII_M Orig	24.38	17.96	12.41	15.26	9.57	16.95	11.14	14.54	8.67	13.55	8.02	12.45	7.05	
CII_Mof64	12.92	12.70	12.46	12.60	12.30	12.58	12.28	12.51	12.15	12.54	12.21	12.44	12.04	
SSEMo_M	24.38	17.96	12.32	15.26	6.05	16.95	9.21	14.54	4.80	13.55	3.97	12.45	3.24	
SSEMo_M Orig.	24.38	17.96	12.32	15.26	6.05	16.95	9.21	14.54	4.80	13.55	3.97	12.45	3.24	

```
3.67*POWER(64,(0.28
      +0.2*(B-0.91)))
    *SCED%/100
=IF(
CII_PMnoSCED<16.001,
SQRT(CII_PMnoSCED),
IF(CII_PMnoSCED<64,
 ((CII_Mof64-4)/48*
  CII_PMnoSCED)+(4
   -16*(CII_Mof64-4)/48),
 3.67*POWER(
   CII_PMnoSCED,
   (0.28+0.2*(B-0.91)))
     *SCED))
```

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Schedule		RVHL: Reuse and Very High Level Language											Inception	
Inception		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
RVHL-M	defau	1.00	0.99	0.98	0.98	0.96	0.98	0.96	0.97	0.94	0.97	0.94	0.96	0.92
RVHL-M	new													

PM same



RVHL Projection Rationales

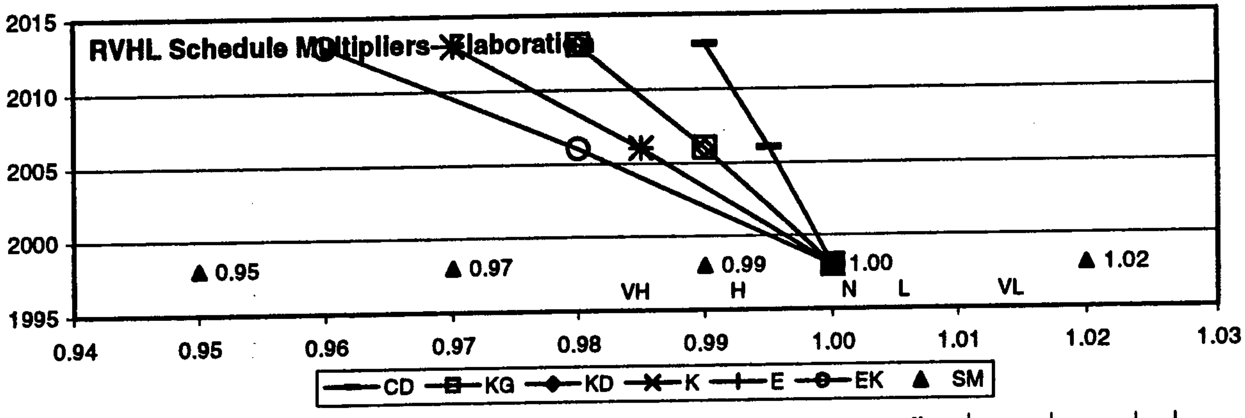
Baseline:	Relatively low current capability and experience in EHART domain (standard 3GL module reuse)
CD:	As indicated under SIZE in the Effort impact analysis, commercial technology and DoD EHART domain initiatives will provide some but not much improvement over standard 3GL module reuse
KD:	Some gains over CD via domain oriented reuse asset identification and decision support
KG:	Some gains over CD via domain oriented prototype applications generation
K:	Complementary gains from KD and KG
E:	Significant gains over CD via domain architecture technology and associated prototype applications generation
EK:	Some complement any gains from E and K
NOTE:	RVHL effects in construction accounted for with regular COCOMOII effort adjustment

App-3-3--RAD Drivers

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Schedule		RVHL: Reuse and Very High Level Language										Elaboration	
Elaboration		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
RVHL-M	default	1.00	1.00	0.99	0.99	0.98	0.99	0.98	0.99	0.97	0.99	0.97	0.98	0.96
RVHL-M	new													

PM same

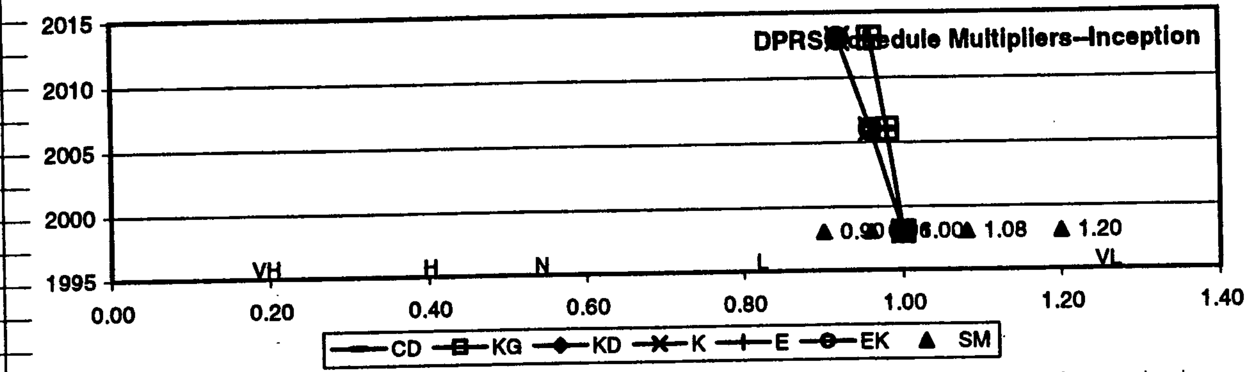


RVHL Projection Rationales (Repeated)

Baseline:	Relatively low current capability and experience in EHART domain (standard 3GL module reuse)
CD:	As indicated under SIZE in the Effort impact analysis, commercial technology and DoD EHART domain initiatives will provide some but not much improvement over standard 3GL module reuse
KD:	Some gains over CD via domain oriented reuse asset identification and decision support
KG:	Some gains over CD via domain oriented prototype applications generation
K:	Complementary gains from KD and KG
E:	Significant gains over CD via domain architecture technology and associated prototype applications generation
EK:	Some complement any gains from E and K
NOTE:	RVHL effects in construction accounted for with regular COCOMOII effort adjustment

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Sched.		DPRS: Development Process Reengineering & Streamlining											Inception	
Driver		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
DPRS-M	default	1.00	0.98	0.96	0.98	0.96	0.96	0.92	0.96	0.92	0.98	0.96	0.96	0.92
DPRS-M	new													
PM same														

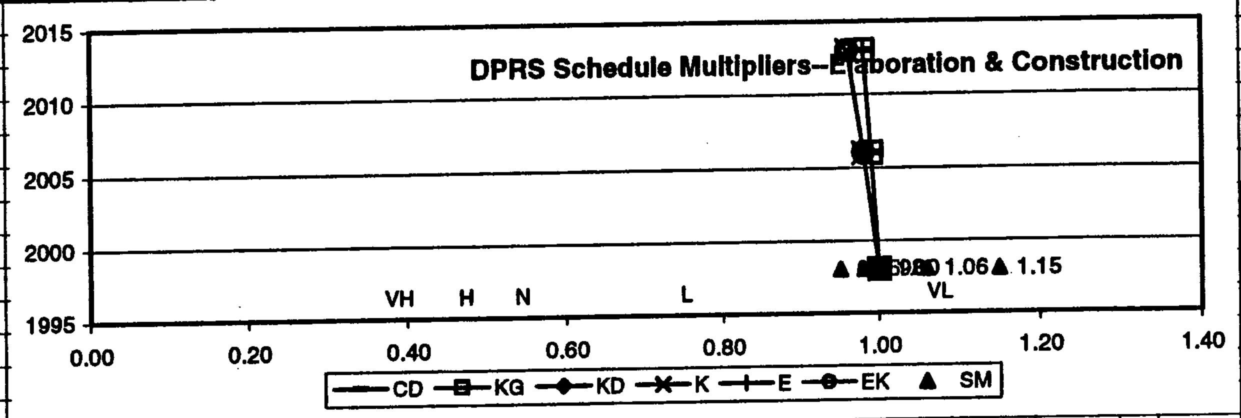


DPRS Rationales

Baseline:	Basic good business practices
CD:	Significant gains via evolution of commercial and DoD best practices
KG:	Same as CD
KD:	Significant gains over CD via knowledge based process model selection, process diagnosis, domain process streamlining
K:	Same as KD
E:	Same as CD
EK:	Same as K

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Sched.		DPRS: Development Process Reengineering & Streamlining											Elaboration & Construction	
Elab&Const		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
DPRS-M	default	1.00	0.99	0.98	0.99	0.98	0.98	0.96	0.98	0.96	0.99	0.98	0.98	0.96
DPRS-M	new													
PM same														



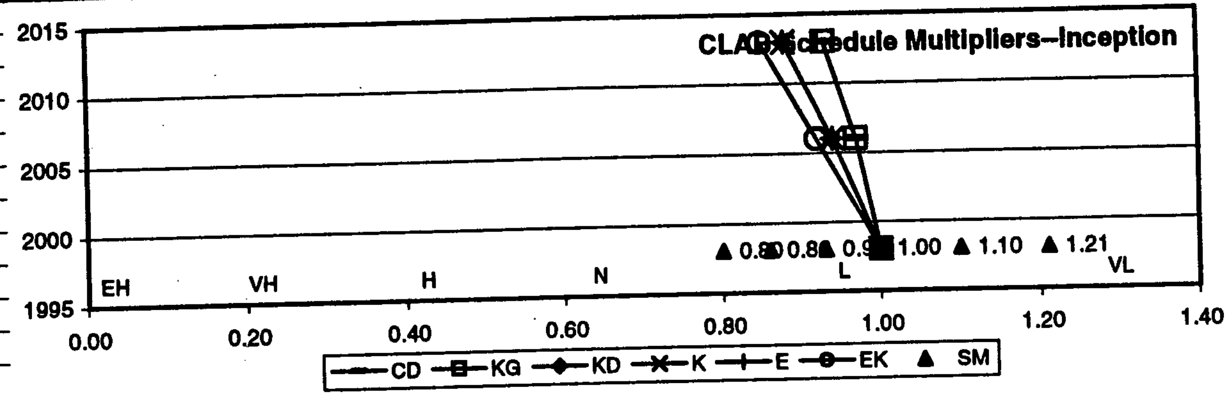
DPRS Rationales (repeated)

Baseline:	Basic good business practices
CD:	Significant gains via evolution of commercial and DoD best practices
KG:	Same as CD
KD:	Significant gains over CD via knowledge based process model selection, process diagnosis, domain process streamlining
K:	Same as KD
E:	Same as CD
EK:	Same as K

App-3-3--RAD Drivers

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Schedule				CLAB: Collaboration Efficiency									Inception	
Inception		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
CLAB-M	default	1.00	0.97	0.93	0.97	0.93	0.94	0.88	0.94	0.88	0.94	0.88	0.92	0.85
CLAB-M	new													
PM same														

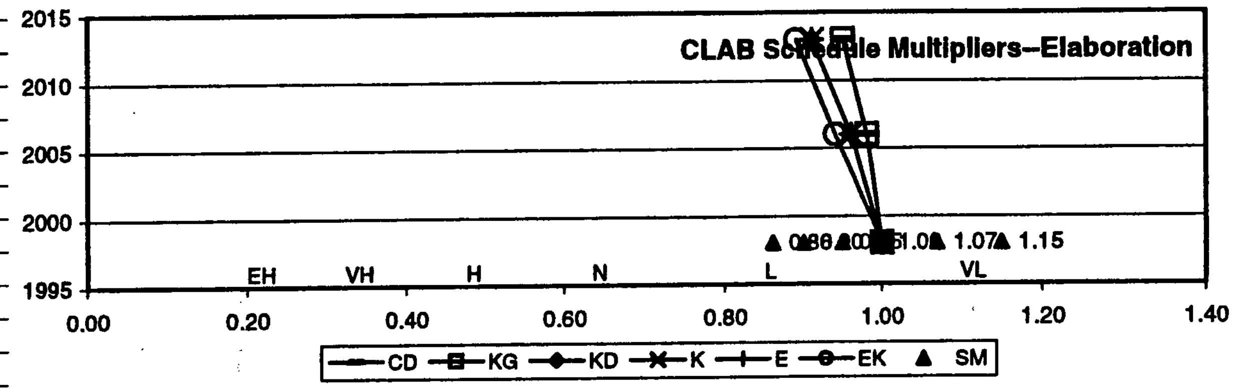


CLAB Projection Rationales

CD:	Significant gains via commercial collaboration technology, DoD Integrated Project Team maturation
KD:	Significant gains over CD via knowledge-based EHART domain, process, architecture decision support; KB increases in effective personnel experience
KG:	Same as CD
K:	Same as KD
E:	Significant gains over CD via advanced collaboration technology; effective EHART domain applications experience
EK:	Major gains over CD via complementary E and K gains

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Schedule		CLAB: Collaboration Efficiency (cont.)										Elaboration		
Elaboration		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
CLAB-M	default	1.00	0.98	0.95	0.98	0.95	0.96	0.91	0.96	0.91	0.96	0.91	0.94	0.89
CLAB-M	new													
PM same														

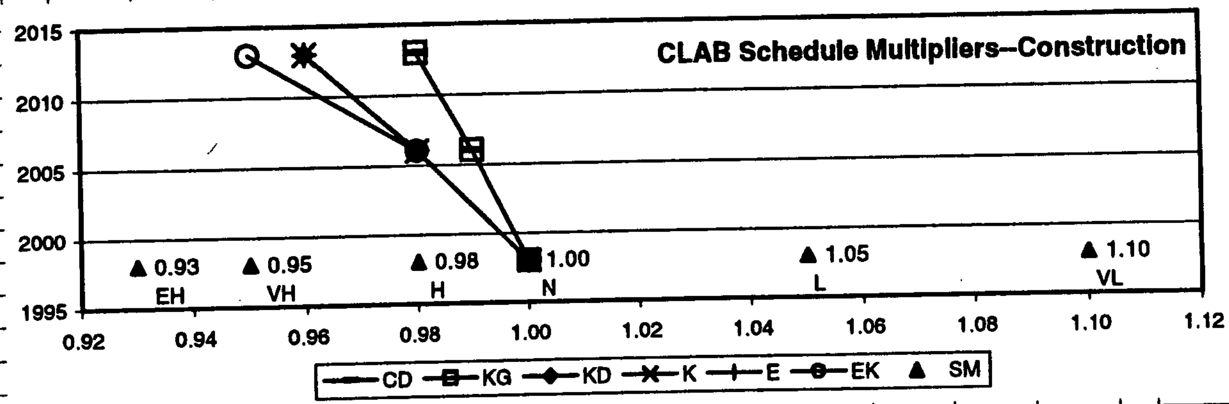


CLAB Projection Rationales (repeated)

CD:	Significant gains via commercial collaboration technology, DoD Integrated Project Team maturation
KD:	Significant gains over CD via knowledge-based EHART domain, process, architecture decision support; KB increases in effective personnel experience
KG:	Same as CD
K:	Same as KD
E:	Significant gains over CD via advanced collaboration technology; effective EHART domain applications experience
EK:	Major gains over CD via complementary E and K gains

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Schedule		CLAB: Collaboration Efficiency (cont.)										Construction	
Construction	Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
CLAB-M default	1.00	0.99	0.98	0.99	0.98	0.98	0.96	0.98	0.96	0.98	0.96	0.98	0.95
CLAB-M new													
PM same													

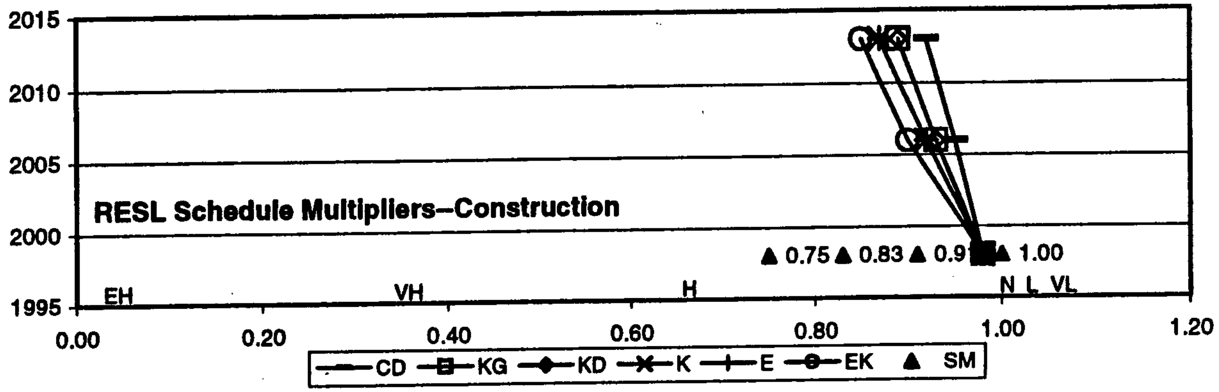


CLAB Projection Rationales (repeated)

CD:	Significant gains via commercial collaboration technology, DoD Integrated Project Team maturation
KD:	Significant gains over CD via knowledge-based EHART domain, process, architecture decision support; KB increases in effective personnel experience
KG:	Same as CD
K:	Same as KD
E:	Significant gains over CD via advanced collaboration technology; effective EHART domain applications experience
EK:	Major gains over CD via complementary E and K gains

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Schedule			RESL: Architecture/Risk Resolution										Construction	
Construction		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
RESL-M	default	0.98	0.95	0.92	0.93	0.89	0.93	0.89	0.92	0.87	0.92	0.87	0.90	0.85
RESL-M	new													
PM same														

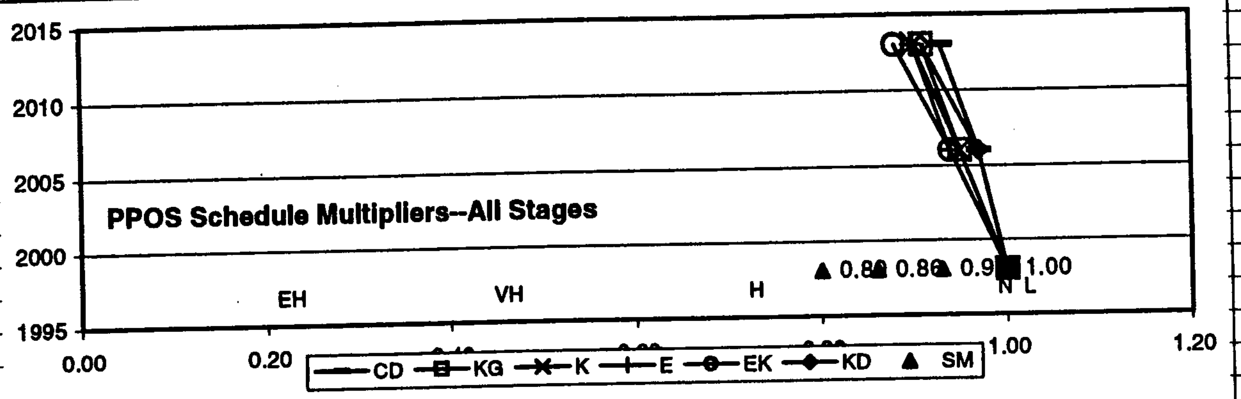


RESL Projection Rationales

	Same rationale as for COCOMO II RESL evaluation; same rating levels
CD:	Significant gains due to open systems technology, commercial OO architecture technology and DoD emphasis on risk management
KG:	Significant additional gains over CD via domain architectures
KD:	Significant additional gains over CD via architecture and risk advisor technology
K:	Complementary gains from KG and KD
E:	Major additional gains over CD via domain architecture, general architecture technology, high assurance technology, and rationale capture
EK:	Complementary gains from E and K

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Schedule		PPOS: Prepositioning Assets							Inception, Elaboration & Construction						
I, E & C		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
PPOS-M	default	1.00	0.97	0.93	0.95	0.91	0.97	0.91	0.95	0.90	0.94	0.90	0.94	0.88
PPOS-M	new													



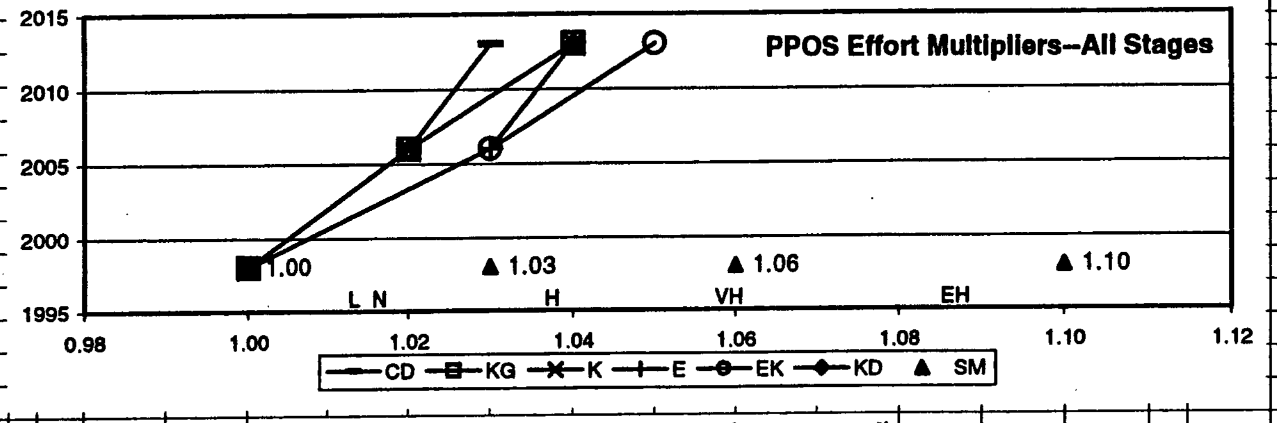
PPOS Projection Rationales

CD:	Significant gains via commercial and DoD asset reuse and product line initiatives
KD:	Some long-range gains over CD via knowledge-based product line process aids
KG:	Same gains over CD via domain architectures and assets
K:	Same as KG in 2006; complementary KD and KG gains in 2013
E:	Significant gains over CD via advanced domain architecture, hyperweb environment support
EK:	Same as E in 2006; complementary E and KD gains in 2013

App-3-3--RAD Drivers

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Effort		PPOS: Prepositioning Assets							Inception, Elaboration & Construction					
I, E & C		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
PPOS-PM	default	1.00	1.02	1.03	1.02	1.04	1.02	1.04	1.02	1.04	1.03	1.04	1.03	1.05
PPOS-PM	new													

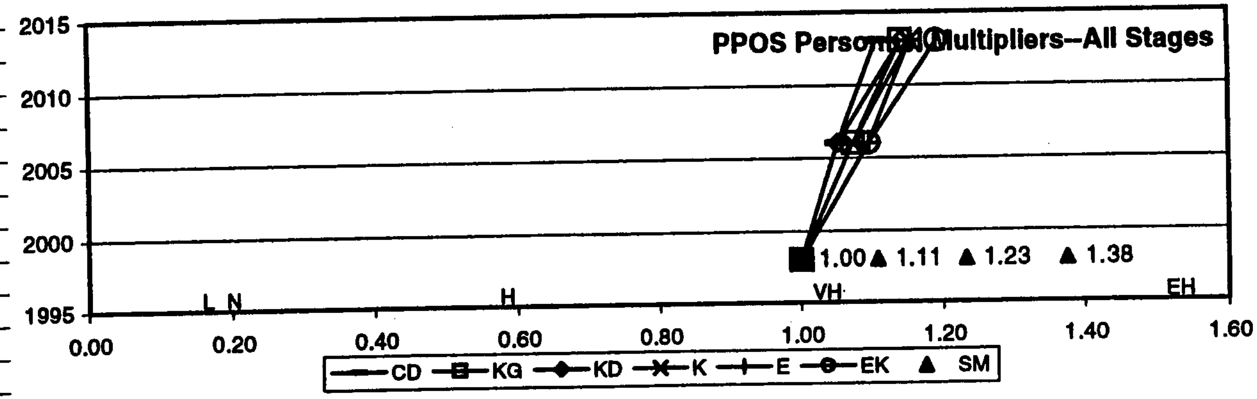


PPOS Projection Rationales (repeated)

CD:	Significant gains via commercial and DoD asset reuse and product line initiatives
KD:	Some long-range gains over CD via knowledge-based product line process aids
KG:	Same gains over CD via domain architectures and assets
K:	Same as KG in 2006; complementary KD and KG gains in 2013
E:	Significant gains over CD via advanced domain architecture, hyperweb environment support
EK:	Same as E in 2006; complementary E and KD gains in 2013

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

FSP		PPOS: Prepositioning Assets							Inception, Elaboration & Construction					
I, E & C		Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
PPOS-P	default	1.00	1.02	1.03	1.02	1.04	1.02	1.04	1.02	1.04	1.03	1.04	1.03	1.05
PPOS-P	new													



	PPOS Projection Rationales (repeated)
CD:	Significant gains via commercial and DoD asset reuse and product line initiatives
KD:	Some long-range gains over CD via knowledge-based product line process aids
KG:	Same gains over CD via domain architectures and assets
K:	Same as KG in 2006; complementary KD and KG gains in 2013
E:	Significant gains over CD via advanced domain architecture, hyperweb environment support
EK:	Same as E in 2006; complementary E and KD gains in 2013

Technology Impact Analyzer -- Individual RAD Schedule Multiplier values and Rationales over time (now, +8 and +15 years; 1998, 2006 and 2013)

Technology Impact Analyzer: Schedule Multipliers -- N
PM=effort(person month) multiplier
M=schedule(months) multiplier

% Effort Inception	14.0	% Effort Elaboration	28.0	%Sched Inception	40.0	%Sched Elaboration	40.0

Baseline(stage) from % allocations and SSE's M=f(PM)

	SchedMult	Baseline(now 1998		CD+8Yr 2006		CD +15Yr 2013		KG+8Yr 2006		KG+15Yr 2013		KD+8Yr 2006		KD+15Yr 2013		K+8Yr 2006		K+15Yr 2013		E+8Yr 2006		E+15Yr 2013		EK+8Yr 2006		EK+15Yr 2013	
		PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M
I	RVHL	1.00	1.00	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.97	0.97	0.94	0.94	0.97	0.97	0.94	0.94	0.96	0.96	0.92	0.92
N	DPRS	1.00	1.00	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.96	0.96	0.92	0.92	0.96	0.96	0.92	0.92	0.98	0.98	0.96	0.96	0.96	0.96	0.92	0.92
C	CLAB	1.00	1.00	0.97	0.97	0.93	0.93	0.97	0.97	0.93	0.93	0.94	0.94	0.88	0.88	0.94	0.94	0.88	0.88	0.94	0.94	0.88	0.88	0.92	0.92	0.85	0.85
E	RESL	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
P	PPOS	1.00	1.00	1.02	0.97	1.03	0.93	1.02	0.95	1.04	0.91	1.02	0.97	1.04	0.91	1.02	0.95	1.04	0.90	1.03	0.94	1.04	0.90	1.03	0.94	1.05	0.88
T	Π	1.00	1.00	0.96	0.91	0.90	0.81	0.95	0.89	0.89	0.78	0.90	0.86	0.81	0.71	0.89	0.83	0.79	0.68	0.92	0.84	0.83	0.71	0.87	0.80	0.76	0.63
I	Baseline I	70.8	9.8	28.6	7.2	9.2	4.9	17.4	6.1	4.1	2.4	24.6	6.8	6.7	3.7	15.2	5.8	3.0	1.9	12.0	5.4	2.3	1.6	9.3	5.0	1.5	1.3
O	New I	70.8	9.8	27.5	6.6	8.3	4.0	16.5	5.4	3.6	1.9	22.2	5.8	5.4	2.6	13.6	4.8	2.4	1.3	11.0	4.6	1.9	1.1	8.2	4.0	1.2	0.8
N	*P(New I)*	*7.3*	*7.3*	*4.2*	*4.2*	*2.1*	*2.1*	*3.1*	*3.1*	*1.9*	*1.9*	*3.8*	*3.8*	*2.1*	*2.1*	*2.8*	*2.8*	*1.8*	*1.8*	*2.4*	*2.4*	*1.7*	*1.7*	*2.1*	*2.1*	*1.4*	*1.4*
E	RVHL	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.99	0.98	0.98	0.99	0.99	0.98	0.98	0.99	0.99	0.97	0.97	0.99	0.99	0.97	0.97	0.98	0.98	0.96	0.96
L	DPRS	1.00	1.00	0.99	0.99	0.98	0.98	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96
A	CLAB	1.00	1.00	0.98	0.98	0.95	0.95	0.98	0.98	0.95	0.95	0.96	0.96	0.91	0.91	0.96	0.96	0.91	0.91	0.96	0.96	0.91	0.91	0.94	0.94	0.89	0.89
B	RESL	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
O	PPOS	1.00	1.00	1.02	0.97	1.03	0.93	1.02	0.95	1.04	0.91	1.02	0.97	1.04	0.91	1.02	0.95	1.04	0.90	1.03	0.94	1.04	0.90	1.03	0.94	1.05	0.88
R	Π	1.00	1.00	0.98	0.94	0.95	0.86	0.98	0.91	0.95	0.83	0.95	0.90	0.89	0.78	0.95	0.88	0.88	0.76	0.96	0.88	0.90	0.78	0.93	0.85	0.86	0.72
A	Baseline E	141.5	9.8	57.3	7.2	18.4	4.9	34.7	6.1	8.1	2.4	49.2	6.8	13.5	3.7	30.5	5.8	6.0	1.9	24.0	5.4	4.6	1.6	18.7	5.0	3.1	1.3
T	New E	141.5	9.8	56.4	6.7	17.5	4.2	34.0	5.6	7.7	2.0	46.7	6.1	12.0	2.9	28.8	5.1	5.3	1.5	23.1	4.8	4.1	1.2	17.4	4.2	2.6	0.9
N	*P(New E)*	*14.5*	*14.5*	*8.4*	*8.4*	*4.1*	*4.1*	*6.1*	*6.1*	*3.8*	*3.8*	*7.6*	*7.6*	*4.2*	*4.2*	*5.6*	*5.6*	*3.6*	*3.6*	*4.8*	*4.8*	*3.3*	*3.3*	*4.1*	*4.1*	*2.8*	*2.8*
C	RVHL	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
O	DPRS	1.00	1.00	0.99	0.99	0.98	0.98	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96
N	CLAB	1.00	1.00	0.99	0.99	0.98	0.98	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.98	0.98	0.95	0.95
S	RESL	1.00	0.98	1.00	0.95	1.00	0.92	1.00	0.93	1.00	0.89	1.00	0.93	1.00	0.89	1.00	0.92	1.00	0.87	1.00	0.92	1.00	0.87	1.00	0.90	1.00	0.85
T	PPOS	1.00	1.00	1.02	0.97	1.03	0.93	1.02	0.95	1.04	0.91	1.02	0.97	1.04	0.91	1.02	0.95	1.04	0.90	1.03	0.94	1.04	0.90	1.03	0.94	1.05	0.88
R	Π	1.00	0.98	1.00	0.90	0.99	0.82	1.00	0.87	1.00	0.78	0.98	0.87	0.96	0.75	0.98	0.84	0.96	0.72	1.00	0.84	0.98	0.74	0.99	0.81	0.96	0.68
C	Baseline C	363.9	14.6	147.3	10.8	47.3	7.4	89.3	9.2	20.9	3.6	126.5	10.2	34.6	5.5	78.4	8.7	15.5	2.9	61.7	8.1	11.8	2.4	48.0	7.5	7.8	1.9
T	New C	363.9	14.3	147.2	9.7	46.8	6.1	89.3	7.9	20.8	2.8	123.9	8.8	33.2	4.1	76.8	7.3	14.8	2.1	61.6	6.8	11.5	1.8	47.5	6.1	7.5	1.3
N	*P(New C)*	*25.4*	*25.4*	*15.1*	*15.1*	*7.7*	*7.7*	*11.3*	*11.3*	*7.4*	*7.4*	*14.1*	*14.1*	*8.0*	*8.0*	*10.5*	*10.5*	*7.1*	*7.1*	*9.0*	*9.0*	*6.6*	*6.6*	*7.8*	*7.8*	*5.7*	*5.7*
	New I+E+C	576.3	33.8	231.1	23.0	72.6	14.3	139.8	18.9	32.1	6.7	192.8	20.7	50.6	9.6	119.2	17.3	22.5	4.9	95.8	16.1	17.5	4.1	73.0	14.3	11.3	3.1
	P(new I+E+C)	*17.0*	*17.0*	*10.0*	*10.0*	*5.1*	*5.1*	*7.4*	*7.4*	*4.8*	*4.8*	*9.3*	*9.3*	*5.3*	*5.3*	*6.9*	*6.9*	*4.6*	*4.6*	*5.9*	*5.9*	*4.3*	*4.3*	*5.1*	*5.1*	*3.7*	*3.7*
	Baseline PM/M	20.7	20.7	11.4	11.4	5.3	5.3	8.1	8.1	4.8	4.8	10.4	10.4	5.2	5.2	7.5	7.5	4.5	4.5	6.3	6.3	4.1	4.1	5.4	5.4	3.4	3.4
	New E+C	505.5	24.1	203.6	16.5	64.3	10.3	123.3	13.5	28.5	4.8	170.6	14.9	45.1	7.0	105.6	12.4	20.2	3.5	84.8	11.6	15.7	3.0	64.9	10.3	10.1	2.3
	P(new E+C)	*21.0*	*21.0*	*12.4*	*12.4*	*6.2*	*6.2*	*9.1*	*9.1*	*5.9*	*5.9*	*11.4*	*11.4*	*6.5*	*6.5*	*8.5*	*8.5*	*5.7*	*5.7*	*7.3*	*7.3*	*5.2*	*5.2*	*6.3*	*6.3*	*4.5*	*4.5*
	CII-OT(I+EC)	576.3	34.1	233.2	25.2	75.0	17.3	141.4	21.4	33.0	8.5	200.2	23.7	54.8	12.9	124.1	20.4	24.5	6.7	97.7	19.0	18.7	5.6	76.1	17.4	12.4	4.5
	(New(I+E+C))	576.3	33.8	231.1	23.0	72.6	14.3	139.8	18.9	32.1	6.7	192.8	20.7	50.6	9.6	119.2	17.3	22.5	4.9	95.8	16.1	17.5	4.1	73.0	14.3	11.3	3.1
		PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M
	CII-OT(E+C)	506.48	24.38	204.55	17.96	65.76	12.32	124.00	15.26	28.97	6.05	175.66	16.95	48.05	9.21	108.83	14.54	21.51	4.80	85.66	13.55	16.37	3.97	66.71	12.45	10.90	3.24
	Sum(New(E+C))	505.5	24.1	203.6	16.5	64.3	10.3	123.3	13.5	28.5	4.8	170.6	14.9	45.1	7.0	105.6	12.4	20.2	3.5	84.8	11.6	15.7	3.0	64.9	10.3	10.1	2.3

N. Appendix 4.

KBSA ADM Evaluation

Prasanta Bose

N.1 Introduction

The KBSA Advanced Development Model developed as part of the Rome Laboratory's Knowledge-Based Software Assistant effort is aimed at improving software development productivity and software quality. The fundamental approach in achieving the above goal is providing automated support that mediates, automates and documents all activities throughout the development lifecycle for both individual developers and teams of developers. The challenge is building such computer-based assistants as elaborated in the KBSA program vision [Green et al., 1983].

The key concept to meeting the above challenge is based on the understanding that the software development is a knowledge intensive activity. Creating large software based systems requires knowledge of the domain (typically multi-disciplinary in nature), the knowledge of the process context, knowledge of existing components and hardware, and personal resources. The KBSA approach is then to provide means for capture and effective use of such knowledge with the goal that such use of such knowledge by the stakeholders will lead to timely production of high-quality software.

Given the above understanding, the key idea in the KBSA approach is exploiting artificial intelligence (AI) concepts and representations to capture and use knowledge. The different types of product specific knowledge are user requirements, system specifications, code, test scenarios and documentation. Process specific knowledge corresponds to the software development plans, resources and status of the project. Some major problems here are: a) integrated usage of the knowledge [Selfridge, 1992] b) managing change and c) managing complexity. Significant progress has been made in addressing the above problems individually [Johnson et al., 1991; Mi et al., 1990; Smith, 1991]. The ADM builds on such advances to provide an integrated set of concepts and tools that address the problems within a single framework. In the following subsections we elaborate on the problems that ADM addresses, and present our approach to evaluating it.

N.1.1 ADM Evaluation: Approach

The focus of this report is on the analytic evaluation (via case studies) of the ADM support concepts. The approach to evaluating ADM involves

a) Identifying the underlying representational constructs for capturing different types of knowledge and operations and functional features based on those constructs in the ADM framework,

b) Performing the usage analysis of such representational constructs and features in software development, and

c) Assessing the utility of the constructs and the features in terms of addressing key software development problems and thereby facilitating software development tasks.

Since any automation concept is targeted to address one or more specific problems arising in the context of software development, we consider the utility assessment in the context of following major problems that arise in development of complex software based systems:

- *Managing complexity.* Complexity of a large software project arises primarily from the complexity of the problem and solution domain and associated space of requirements and design decisions. Due to the complexity, very few stakeholders have a complete understanding of the system. Such global understanding is critical in identifying and resolving conflicts and interactions between decisions, developing a coherent and integrated design, reducing risks arising from uncertainties, and evolving the system as requirements change.
- *Supporting coordination.* Most large-scale systems involve multiple stakeholder communication and decision making. Such stakeholder interactions may range from same-time same-place interactions to different-place different-time interactions. In all such cases, due to the dependencies between the decisions, coordination is required to ensure proper flow of information to relevant parties.
- *Change management.* Change is an essential attribute of all software projects. Changes taking place in requirements and design decisions must be propagated and their effects analyzed to determine how existing decisions are affected.
- *Automation.* In software development there are large numbers of routine tasks that are well understood but tedious to perform. Automating routine tasks leads to improved quality of the design (by reducing the errors introduced by manual steps) as well reduction in the development time. Moreover the automation of best practices can yield significant improvement in achieving desired quality goals.

For a complementary feature-by-feature evaluation of the ADM, see [Fawcett et al., 1997].

N.1.2 ADM Key Ideas and Usage Model

The key ideas underlying the ADM model are:

1. Complexity Management via abstract representations of requirements and design, automation for process enactment, and consistency between work products via critics. The abstract representations capture knowledge relevant to specific concerns. The abstract processes capture knowledge on the process steps, preconditions capturing dependencies on other steps and resources, and effect on the product representations [Mi et al., 1990]. Critics check for constraint violation, monitor dependencies and notify via creation of tasks that resolve such violations.
2. Automation of best practices. This is done via critics that encode transformation knowledge as well as knowledge on issues that arise when best practices are not followed [Johnson et al., 1991].
3. Coordination support. The support is provided via critics that capture dependency knowledge and notify the stakeholders (users) via updating the agenda of tasks.

The two fundamental modes of using ADM are i) Explicit process-driven control – in this mode the software engineering activities are structured and managed by a specified process, which specifies the activities, the actors and an ordering on the activities. ii) Independent control – in this mode, the project members act independently and hence the process remains implicit. Our evaluation study was based on a limited version of ADM that did not provide support for the first mode.

N.1.3 Outline

To identify the ADM support features and understand their usage in software development we first exercised ADM on two medium sized software development problems involving automated banking services and Automating gas station services. Section N.2 reports on the object oriented analysis (using UML) and usage models of

ADM that resulted from such a study. Section N.3 presents the usage analysis of ADM in terms of use-cases and sequence diagrams that elaborate the use-cases. Section N.4 asks the questions on how effective the support concepts and functional features are for improving software development productivity by analyzing them in the context of the problems articulated in section N.1. Section N.5 provides our overall summary evaluation of the ADM in terms of its potential impact on software cost and schedule.

N.2 Object Oriented Analysis & Modeling of ADM Support Elements: ADM Artifact Meta Model

The support concepts in ADM are based on the insight that the engineering of complex software based systems require creation and usage of different models [Rumbaugh et al., 1991] that allows separation of concerns and decomposing the problem to manage complexity. Moreover the models aid in expressing design decisions and visualizing their effects. The KBSA/ADM environment provides support for capturing and relating three distinct models: a) Decision rationale model – modeling business cases, decisions and rationale for them b) Conceptual model for modeling terms and requirements c) Specification model for modeling requirements specification and design.

In a development activity the above models may be created in fragments via activity sessions. This gives flexibility in organizing one's work and also flexibility in structuring the model space. ADM recognizes such a need and allows creation of sessions and structuring sessions based on the model elements called topic types that get instantiated and manipulated as views. The structure of interactions with the ADM environment follows the model-view controller pattern [Krasner et al., 1988] where a model is defined by a set of topic instances and views constitute a working view on a topic instance.

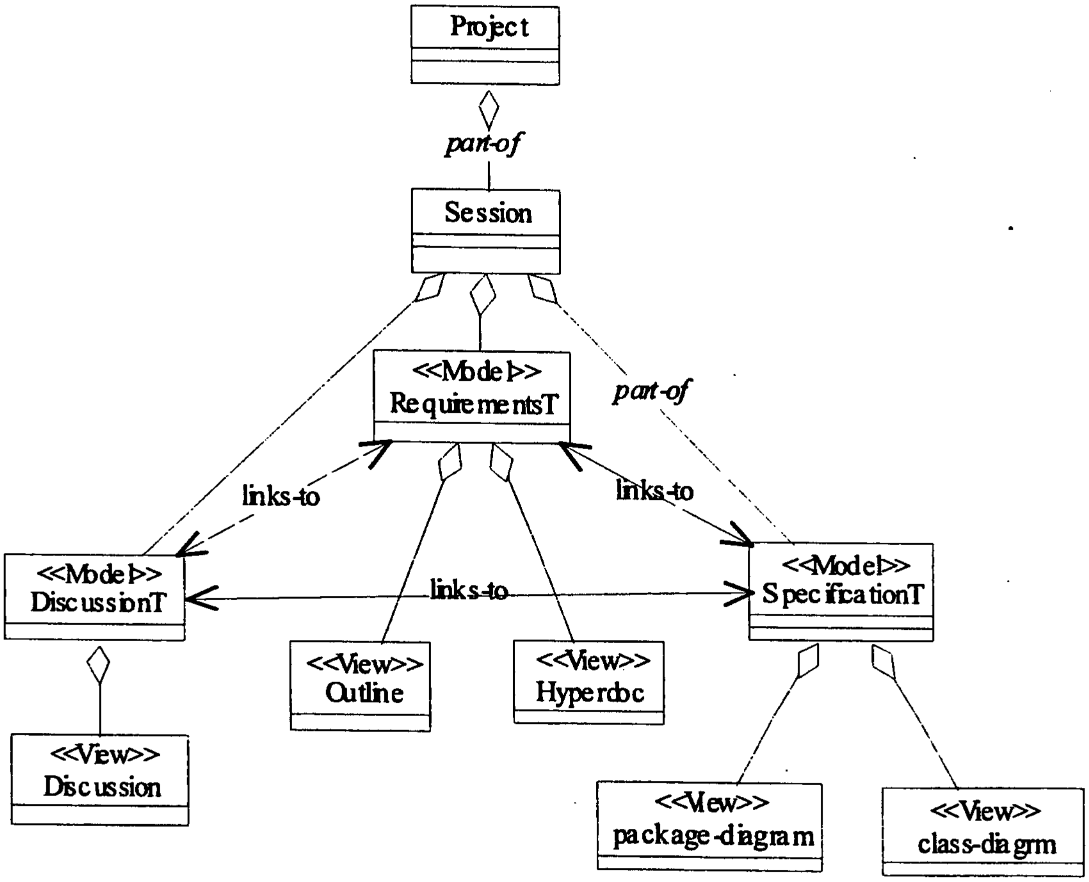


Figure N-1: The KBSA/ADM meta-model showing the models and views created in a software development activity and managed by ADM.

Figure N-1 shows the conceptual model of ADM captured as an UML class diagram. As shown in the figure, a project consists of one or more session objects, a session consists of one or more topic instances (of model stereotype) and a topic instance consists of one or more views. A topic in ADM is modeled as a <<Model>> stereotype and a view is modeled as <<View>> stereotype in UML.

The complexity of providing tools that aid in capture and management of the above categories of models (the topics) is addressed in ADM by a divide and conquer strategy. The ADM support system is a composition of the following tools:

- ***RASE*** - for creating and managing the requirements model elements as well as discussion topics,
- ***ALE*** - for graphical capture and evolving of the object oriented specification models, and
- ***IPSE*** - for graphical capture, editing and enactment of process plans.

The following subsections model the key view specific representation constructs in ADM as first class objects and describe the operations on them relevant to capture of the knowledge.

N.2.1 Requirements Document

The requirements model is informally captured as a set of sentences in English and represented in a tree structured format. The tree-structured format allows two distinct views of the requirements model: a) Hyperdocument view, and b) Outline view or taxonomy view. A metamodel of the requirements model is shown in Figure N-2. The metamodel is represented as a package consisting of two elements that model the normal outline view document and the hyperdocument view. The meta-model shows the key operations.

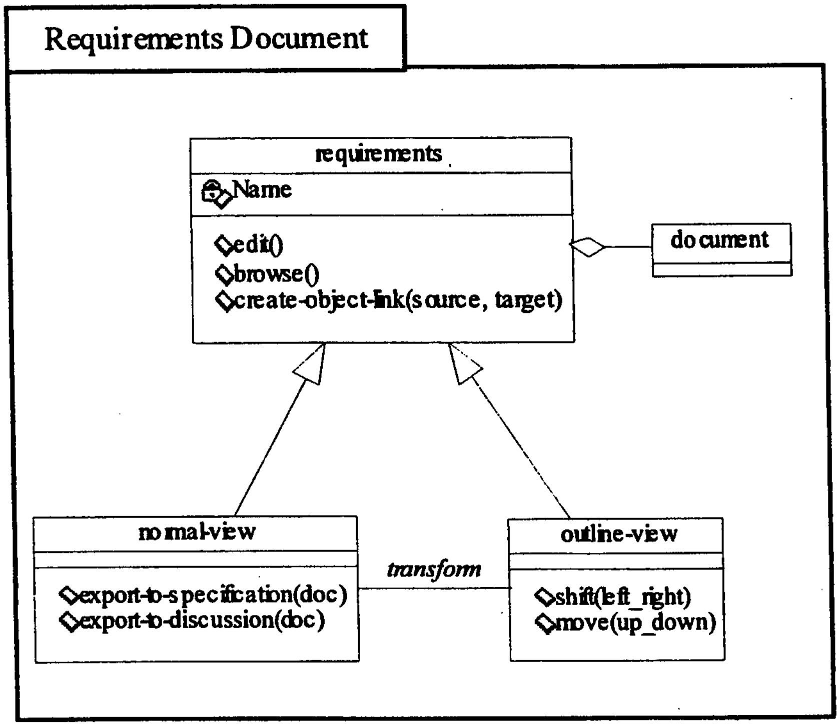


Figure N-2: The requirements document model artifact and the views associated with the model formalized in UML.

N.2.2 **Discussion Artifact**

Figure N-3 shows the discussion artifact model consisting of the root class called 'discussion node' that gets specialized by the 'Argument', 'Issue' 'Position', 'Decision', 'Requirement' and 'Assumption' classes. All classes have two attributes - the discussion element 'name' and 'description' for capturing the content of the element. The key operations supported by each class are for creating links between the discussion nodes in a discussion fragment instance as well as 'Create ObjectLinks (hyperlinks) to other artifact instances.

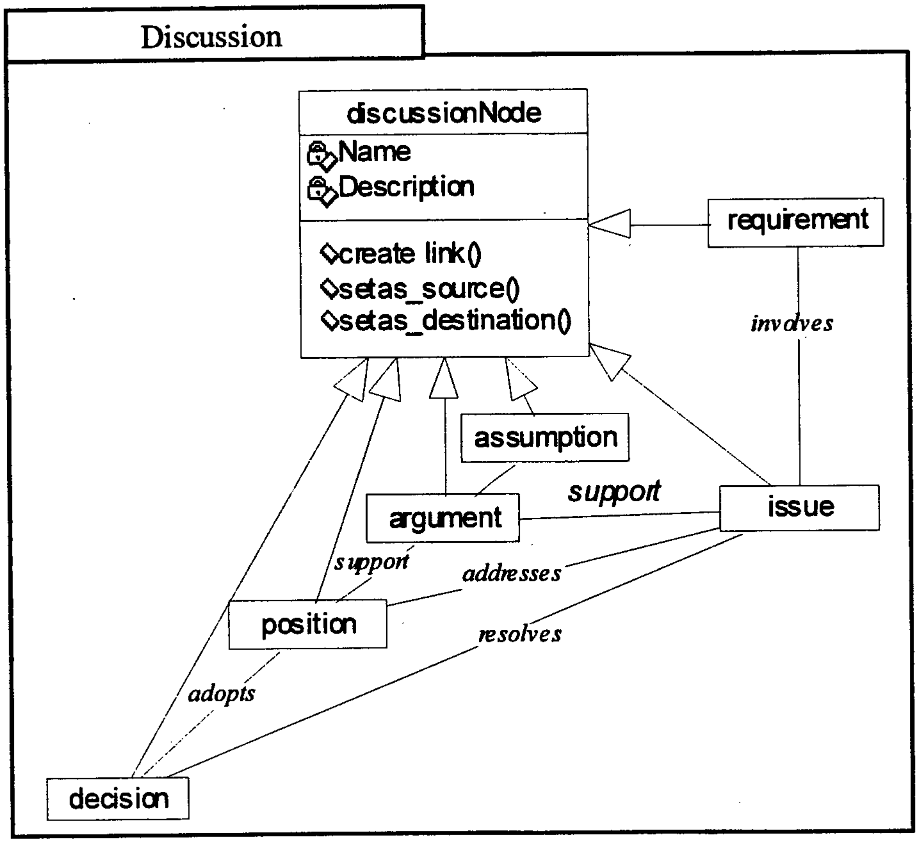


Figure N-3: The discussion model showing the discussion artifacts and their relationships.

N.2.3 Specification Artifact

The specification artifact model (the specification package) in Figure N-4 shows the structure and behavior of the artifact. The structural elements in a Specification are a) 'package' with attribute Name and methods 'Create Package' and 'Create Relation' and b) class diagram The package 'Package Diagram' contains class 'Class Diagram' which in turn has various methods like 'Create Method', Create Attribute' etc.

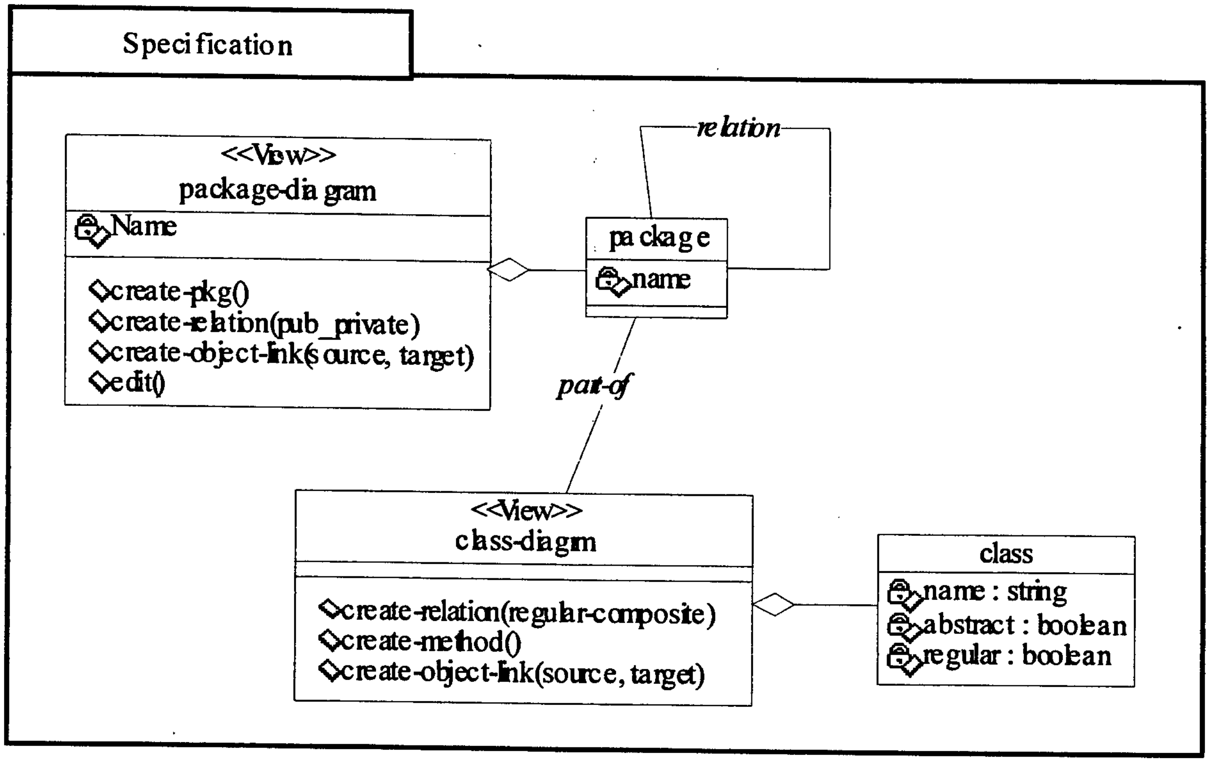


Figure N-4. The specification model for capture of the abstract design knowledge in terms of packages and classes.

The ALE tool in the ADM prototype provides support for package and class diagrams. The package and class representations supported in ALE are a limited form of the UML concept. A package is used as an abstraction of a group of classes. In the ALE diagrammer, lines between class objects are used to represent inheritance relationships. The ALE diagrammer also allows capture of composition relation where a class is an aggregation of two or more objects.

N.2.4 Critics

ADM provides expert knowledge based assitance via critics. Critics encode design knowledge that are used to check for integrity of the representation created by the user (directly or indirectly via application of tools such as compilers). For example, if a class is deleted and the corresponding associated relationship is not deleted, the ALE critic sends a message to the KBSA session manager. The session Manager then adds a resolution to the process plan that requires the user to either delete the dangling relationship or add the class back again. The following are major types of critics supported in ADM:

- Content critic – evaluates the package for its correctness (primarily syntactic checking).
- Task completion critic
- Cohesion and coupling critic

ADM currently does not provide any capability to develop new or edit existing critics.

N.3 ADM Use Case Analysis

ADM can be used to support specific tasks in the software life cycle. A functional evaluation of the ADM concepts and support tools can be performed via use-case analysis [JAC92]. Figure N-5 shows the use-case diagram that captures the relevant usages of the ADM system, their inter-relationships and relationships with the end actors (or users). The following subsections elaborate on each of the use cases and model them using sequence diagrams. The sequence diagram captures the relevant view (e.g. discussion) specific support element and the operations performed. Given such a model that provides a detailed view of each use-case, it is then possible to evaluate how each operation is supported effectively as well as the reduction in the effort involved in performing the specific task. We consider the requirements engineering and design of an ATM for a bank system in describing the models. Moreover in describing the scenarios we consider actions at the conceptual modeling level as opposed to specific menu choices and button clicks.

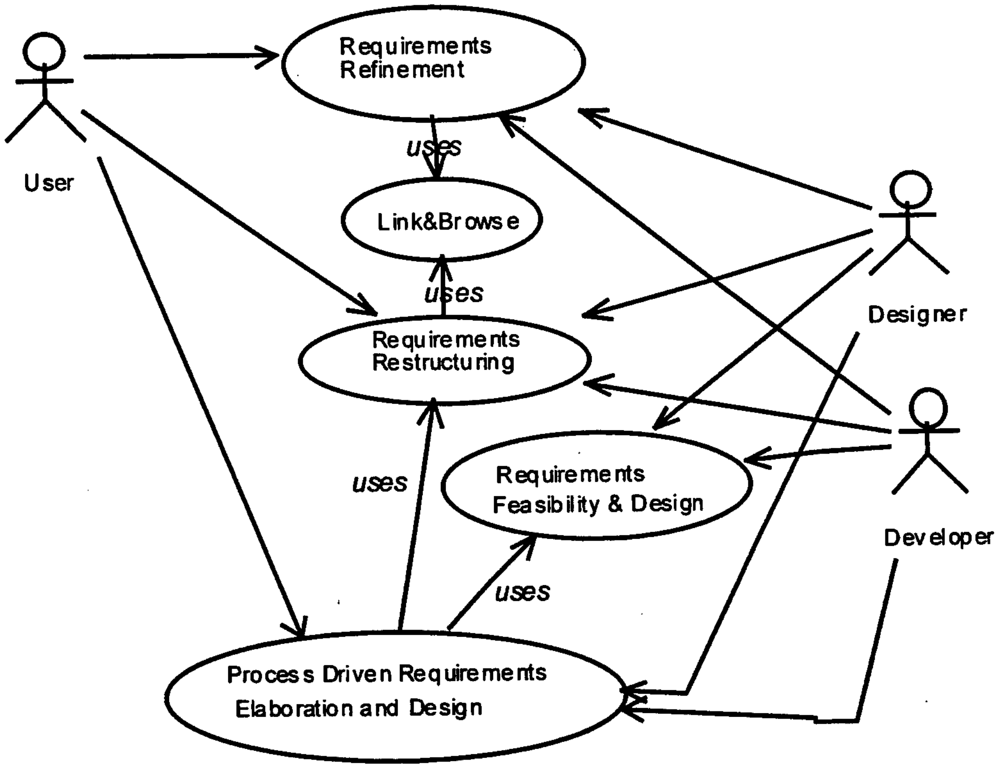

Requirements Refinement
User
uses
Link&Browse
uses
Designer
Requirements Restructuring
uses
Requirements Feasibility & Design
Developer
uses
Process Driven Requirements Elaboration and Design

Figure N-5. Use case diagram showing the possible usage of the ADM tool

N.3.1 Requirements Refinement – Discussion Driven

In the initial phases of the software development life cycle, the requirements identified in the requirements document may need to be refined through discussion and design. Figure N-6 shows the ADM artifacts involved in the requirements refinement process when developing the requirements for the ATM bank example.

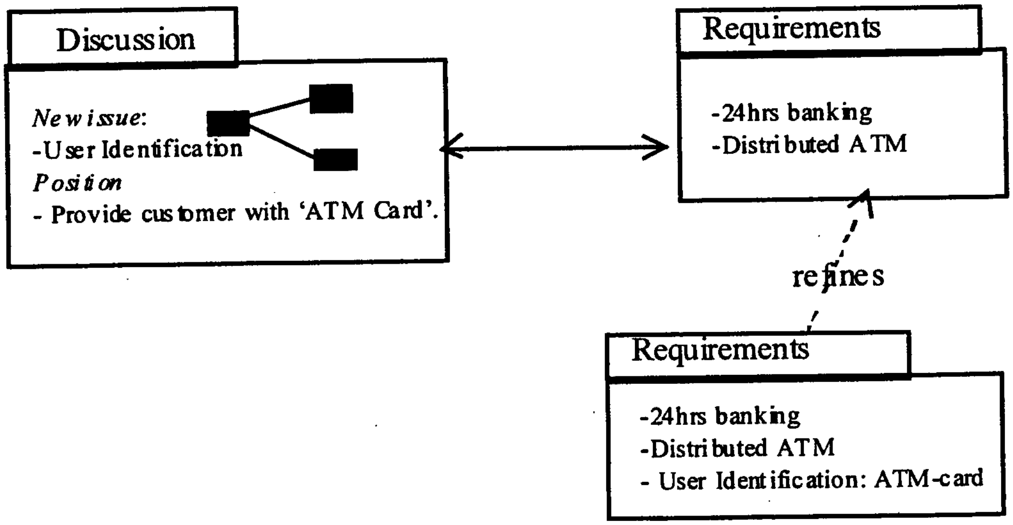


Figure N-6: ADM artifacts involved in requirements refinement.

The detailed scenario of the interaction between the artifacts is described by the sequence diagram in Figure N-7.

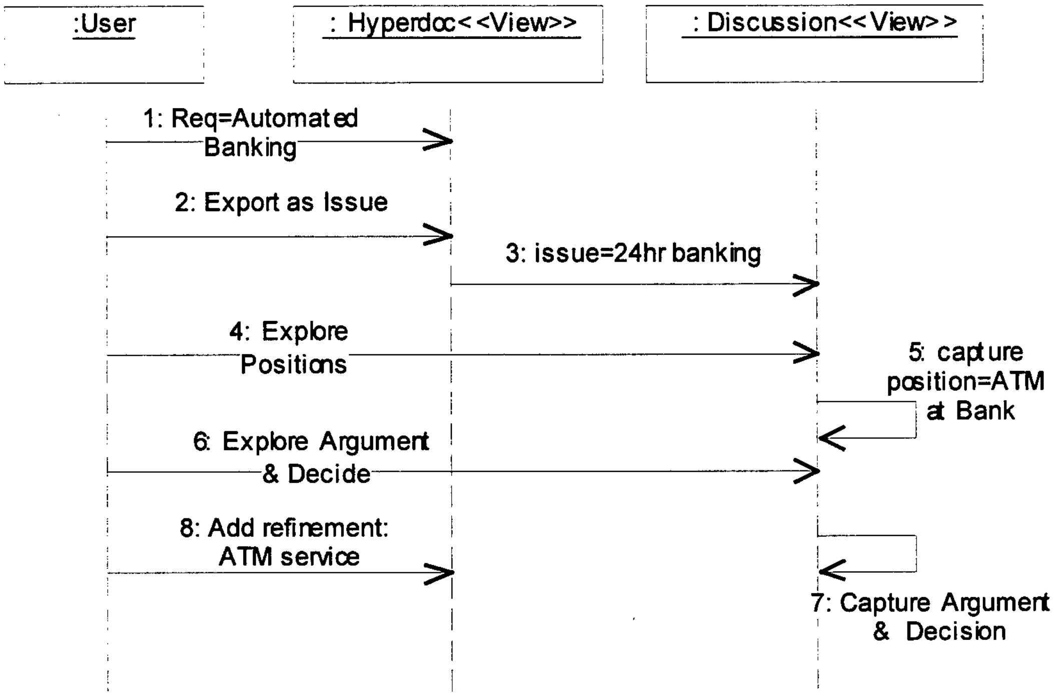


Figure N-7: Sequence diagram showing the artifact interactions and user actions in requirements refinement.

As can be observed from the sequence diagram, the core (or minimum) set of steps involved in the refinement activity consists of the user: a) exporting a requirements document artifact as an issue, b) navigating the discussion graph, c) exporting a discussion artifact (chosen position) as a refinement to a requirement document.

N.3.2 Requirements Restructuring

Discussions on requirements may create arguments and decisions that lead to restructuring (evolution) of the requirements. ADM facilitates such as a process by providing representations and operations to capture the in-process artifacts. Figure N-8 shows the ADM views created and their relationships explored in such an activity.

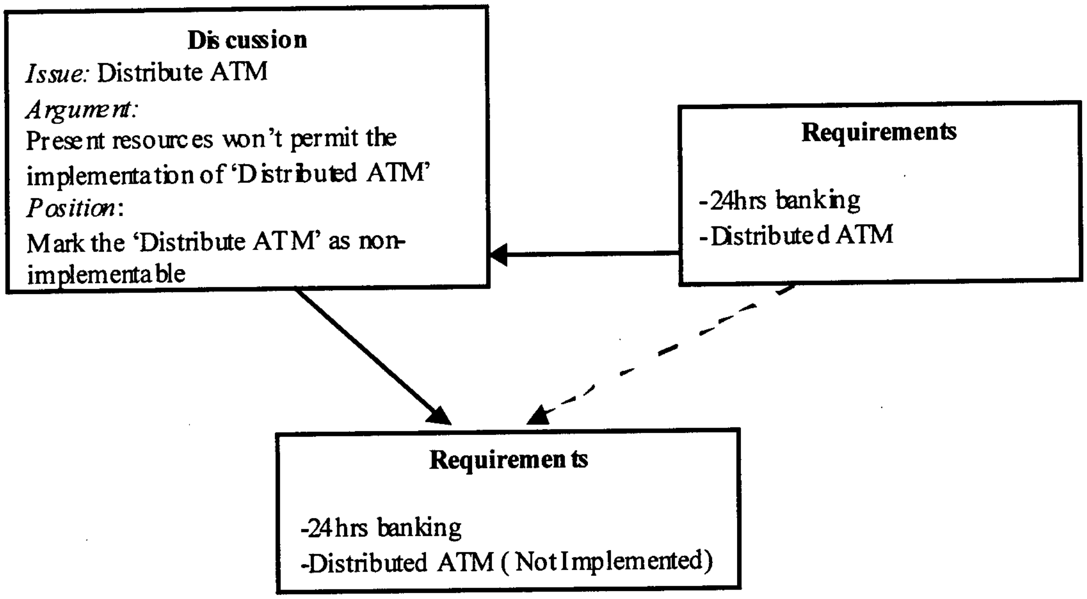

Discussion
Issue: Distribute ATM
Argument:
Present resources won't permit the implementation of 'Distributed ATM'
Position:
Mark the 'Distribute ATM' as non-implementable
Requirements
-24hrs banking
-Distributed ATM
Requirements
-24hrs banking
-Distributed ATM (Not Implemented)

Figure N-8 : The ADM artifacts supporting capture of in-process knowledge when doing requirements restructuring.

The basic scenario involves the following sub-steps:

- Capture the issues
- Mark the issues that cannot be implemented
- Navigate to the requirement document to identify the requirements that raised these issues
- Mark the requirements as Non-Implementables.
- Link the marked requirements to the argument objecting the position in discussion view

N.3.3 Requirements Feasibility Analysis via Design

In the initial stages of requirements decision making, feasibility of some of the requirements decisions need to analyzed to avoid costly backtracking. The requirement feasibility analysis can be focused by the discussion, which identifies those elements in the requirements whose feasibility is of concern to specific stakeholders. Figure N-9 shows the ADM artifacts that aid in capture of the artifacts considered during the activity.

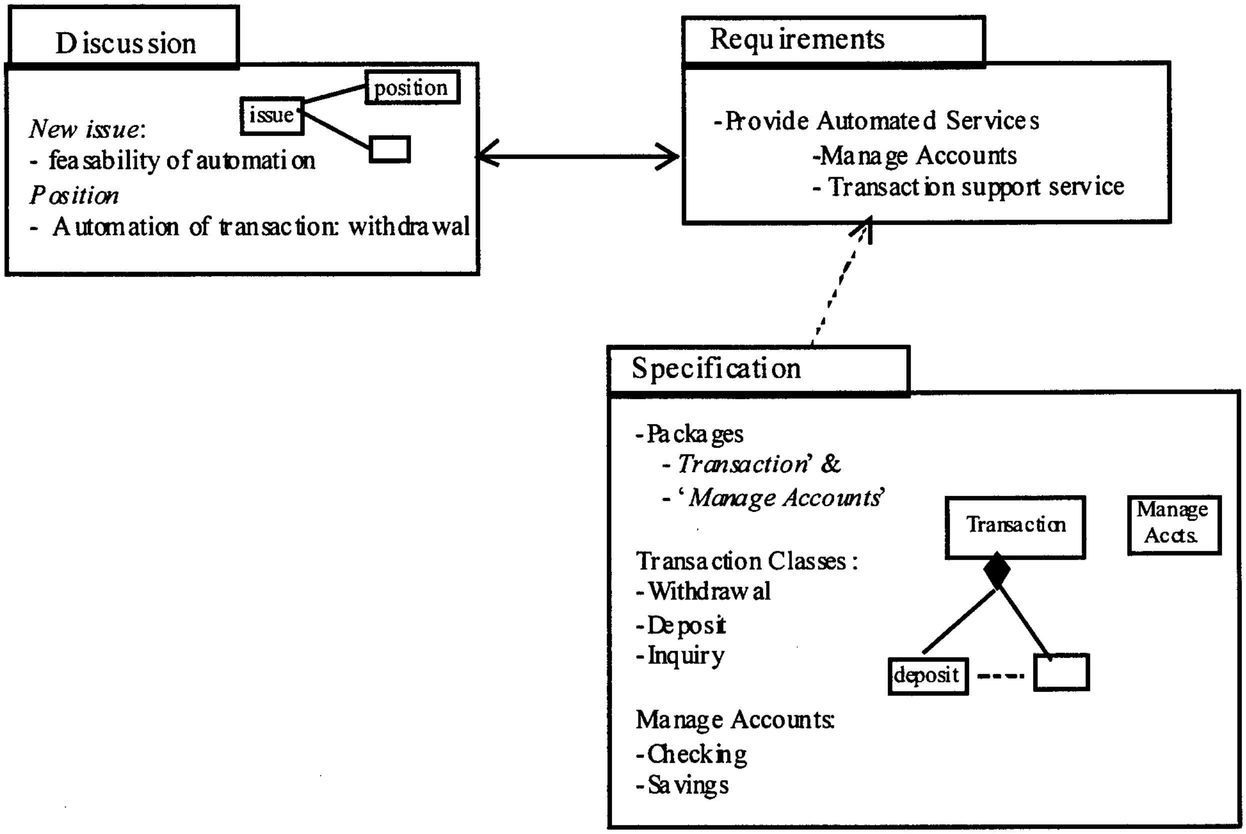


Figure N-9: The ADM artifacts involved in requirements feasibility analysis via design.

A typical scenario elaborating the steps involved in such an activity is shown in Figure N-10. The key steps here are:

- Capture the Issue (I) (whose body specify the requirement).
- Capture the position (of the form “realize using position x”).
- Map requirements to specification
- Elaborate the specification
- Map results of specification modeling to discussion

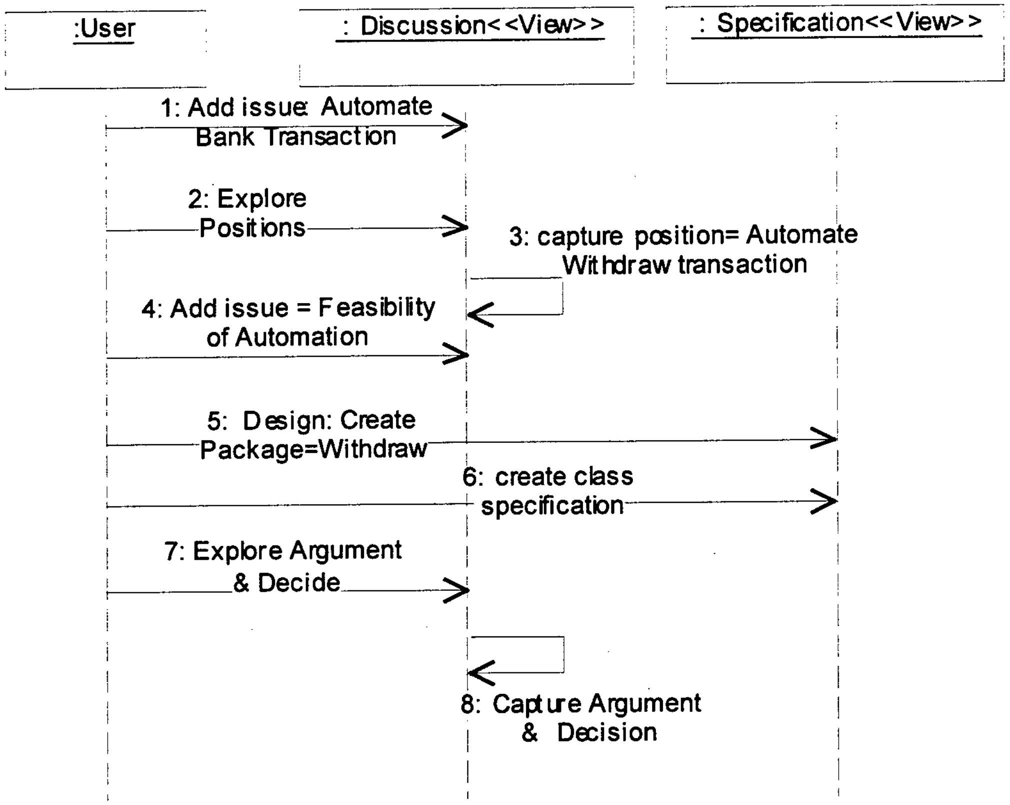

:User
: Discussion<<View>>
: Specification<<View>>
1: Add issue: Automate Bank Transaction
2: Explore Positions
3: capture position= Automate Withdraw transaction
4: Add issue = Feasibility of Automation
5: Design: Create Package=Withdraw
6: create class specification
7: Explore Argument & Decide
8: Capture Argument & Decision

Figure N-10: The sequence diagram elaborating the feasibility analysis usage of ADM.

N.3.4 Position/Option Exploration

In this activity, positions introduced in the discussion process get analyzed via specification creation. ADM supports capture of the discussion elements the specification elements involved in the activity as shown in Figure N-11.

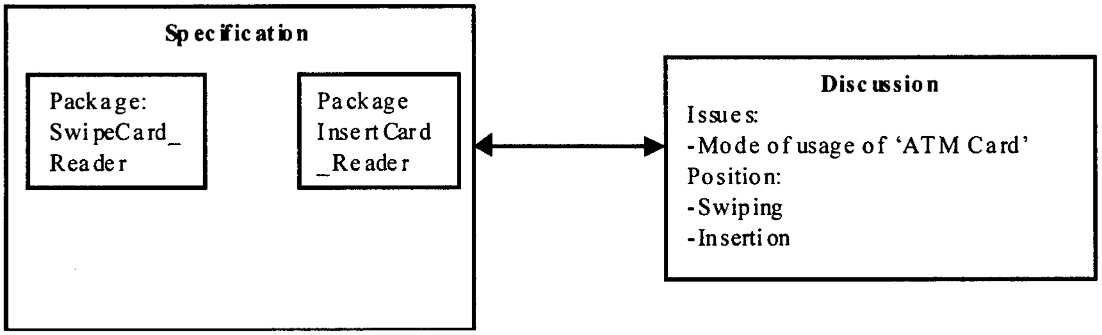


Figure N-11: The ADM artifacts participating in capture of knowledge that support position/option exploration via specification analysis.

A scenario for such an activity would involve:

- Create an issue
- Capture discussion on an issue and the (multiple) positions pertaining the issue
- Create package diagram for each position and class diagram model of the position
- Link individual package to a position

N.3.5 Linking and Navigation

The Object linking capability, which provides the feature of traceability is provided in the KBSA/ADM. The following sequence of steps illustrates the navigational capabilities.

Linking :

- Create any two views such as 'hyper document' and ' discussion'
- Select any entity as source in one view and as target in the other view
- Create object link

Navigation:

- Select one of these views such as 'hyper document'
- Click on the entity which is set as link
- Other view, which is 'discussion' is popped up, as it is linked to the clicked entity of the 'hyper document' view

The navigation is *bi-directional*. It can be done from any of the views to the other views. Figure N-12 schematically shows the creation of the links between an artifact in the discussion view (Issue = 24hrs banking) and an artifact in the hyper-document view (Requirement = necessary to have 24hrs banking).

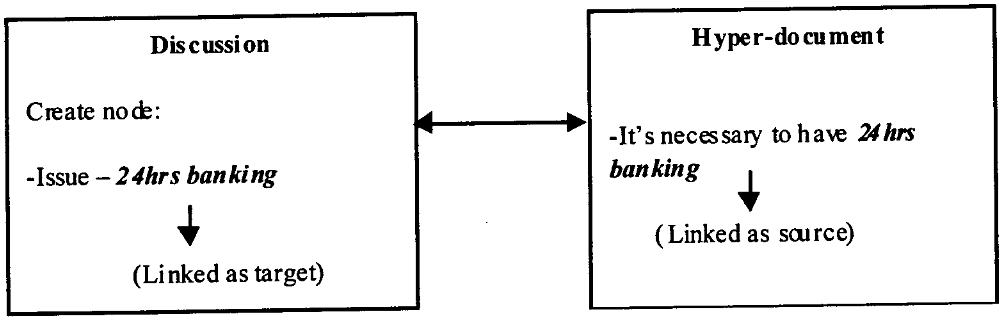


Figure N-12 : Linking elements in the discussion view with elements in the hyper-document view.

N.4 Evaluation of ADM: Effectiveness and Utility

The evaluation of ADM was restricted only to the tools for capture of requirements and design representations. We were unable to run and exercise the process capture and enactment tool. The key strategy in our evaluation effort was to exercise the toolset as a whole as opposed to just evaluating each tool independently. The basis for such a strategy was a) most of the tools were independently developed outside of the ADM project (e.g. REMAP [RD92]) and hence have been evaluated in the course of the research performed on the tool. b) scalability and applicability to engineering of complex software systems that involve multiple stakeholders require coping with such concerns as complexity management, interoperation between the representations, coordination, sharing of models and workflow. We assumed that the primary leverage of ADM concepts and support was in providing such capabilities via integration and evolution of an existing set of components that provided specific capabilities.

The above evaluation goals were achieved in two steps: a) Understanding the representational constructs underlying each tool and the integrated usage of the constructs to support various software engineering activities – such analysis provided a measure of the effectiveness of the representational constructs, and b) Exercising the ADM tool in requirements engineering.and design of medium size systems such as the ATM for a bank and automated gas pump. The objective of the exercise was to evaluate the utility of ADM in managing complexity of medium sized projects. Below we list the observations made from conducting the evaluation study.

- ***Evaluation of ADM framework and integration concepts independent of the context of Usage.*** The ADM approach to integration of components based on the model-view controller architectural style has several advantages and matches very well to support the requirements for scaleable distributed software engineering. The key observations are:

  a) **Management of complexity.** The concept of working in a session characterized by first class view objects managed and controlled by view representation specific tools (the IPSE, the ALE, etc.) makes it possible to effectively capture and manage knowledge relevant to specific software development concerns.

  b) **Flexible evolution.** The style also allows flexible evolution by allowing other view specific components to register and make updates to the model.

  c) **View integration.** The concept of a view in such an approach is to be distinguished from a database view where the view corresponds to a model of a query to the database. The model in the ADM is just the composition (or union) of the views. A major problem in such a multiple view driven software development is ensuring global integrity across the views. The concept of critics could serve an useful role here but ADM fails to effectively exploit such a concept.

  d) **Persistence management and non-intrusive model update.** The use of the object persistence feature provided by the object-store management system and the concept of change in view to be propagated to the model is very effective in non-obtrusive propagation of change in the view to change in the model. The user does not have to take extra steps to save changes to the model.

e) **Asynchronous collaboration.** The nature of collaboration supported in ADM is primarily process driven. A major weakness of ADM is that it does not provide means for product representation driven asynchronous collaboration. The weakness can be addressed by providing stronger support for critics that add specific collaboration goals based on the product models created or changed asynchronously by the individual stakeholders.

- ***Evaluation of ADM tools: Editors for Knowledge Capture, Linking and Transformations.*** The ADM concepts and framework can be understood and evaluated from the viewpoint of software design and development as knowledge-based debugging. The project deliverables are generated via model editors. In such a conceptualization, the evaluation of ADM is based on evaluating the high-level editors for capture, linking, transformation and management of models of the product manipulated in the early phases of the software life cycle. The utility of the individual view specific captured models can be seen from the ADM use-case analysis (in section 3) that shows how changes and refinements in one model can be used to focus changes in another view. The key evaluation results are:

f) ***Hyperdocumentation.*** The major goal of the hyperdocument editor is to provide constructs for capture of loosely structured documents, with links that enable the user to navigate to various other project deliverables. The ADM environment provides a limited editor in terms of kinds of hyperdocuments and their structuring. The linking capability provided by ADM is very strong. Given the current progress and acceptance of standards for hypertext based on HTML (and XML), a major concern is compatibility of ADM hyperdocument representation with such standards.

g) ***Rationale Capture - documentation.*** The discussion view (based on REMAP [Ramesh et al., 1992]) provides an effective means to graphically capture rationale fragments. The graphical approach makes use of icons and is very easy to use. One major problem is the complexity of book-keeping of such rationale structures resulting from the expressivity of the rationale representation (too many node types and relations). Developing automated support for propagation of change and dependency structure analysis based on such representations and that scales-up is very difficult.

h) ***Design Representation.*** The goal of the design representation is to capture design specifications that can be automatically transformed into C++ code. This is a very difficult task. The ALE representations used to capture specifications is very limited and supports generation of only C++ header files. Current technologies (Rational Rose, Visual Basic, Visual C++) have made significant progress in code generation from high-level object models and interface specifications. The challenge is providing automated assurance of global requirements or properties without compromising scalability.

- ***Evaluation of ADM tools: For analysis.*** A major weakness of the current ADM support framework is the lack of tools for analysis of the software views and models being captured in the representations. There is very little analysis via the use of pre-defined set of critics.

N.5 Summary Evaluation of the ADM

1. The ADM environment framework has good technical concepts (model-view controller architecture; process-driven environment; persistent object base).
2. These concepts have several critical issues regarding the feasibility of their use on large, complex projects (scalability; overconstraining people-collaboration processes).
3. The ADM was not fully-enough implemented to resolve these issues.
4. Much of the ADM has been overtaken by commercial technology (e.g., Rational Rose).
5. The ADM has some good concepts such as the use of critics for software project decision assistance or conceptual debugging.

6. For the foreseeable future, KBSA technology will have more impact on software project costs and schedules if pursued in terms of specialized tool enhancements of commercial environments (e.g., domain-specific application generators; critics or software project decision aids), rather than in terms of alternative environments.

N.6 References

[Fawcett et al., 1997]. J. Fawcett. B. Brunk, K. Ganesh, and U. Parvate, "Evaluation of a KBSA Advanced Development Model," Syracuse University Report, May 1997.

[Green et al., 1983]. C. Green, D. Luckham, R. Balzer, T. Cheatham and C. Rich, "Report on a Knowledge-Based Software Assistant", RADC-TR-83-195, August 1983.

[Jacobson et al., 1992]. I. Jacobson et. al., "Object-Oriented Software Engineering", Addison Wesley, 1992.

[Johnson et al., 1991]. W. L. Johnson, M.S. Feather and D. R. Harris, "The KBSA Requirements Specification Facet: ARIES", Proceedings of the 6th KBSE Conference, Syracuse, 1991.

[Krasner et al., 1988]. G. E. Krasner and S. T. Pope, "A cookbook for using the model-view controller user interface paradigm in Smalltalk-80", Journal of Object-Oriented Programming, 1(3):26-49, September 1988.

[Mi et al., 1990]. P. Mi and W. Scacchi, "A Knowledge-Based Environment for Modeling and Simulating Software Engineering Processes", IEEE Transactions in Knowledge and Data Engineering, Sept. 1990.

[Ramesh et al., 1992] B. Ramesh and V. Dhar, "Supporting Systems Development by Capturing Deliberations During Requirements Engineering," IEEE Trans. SW Engr., June 1992, pp. 498-510.

[Rumbaugh et al., 1991]. J. Rumbaugh, et. al, "Object-Oriented Modeling and Design", Prentice Hall, 1991.

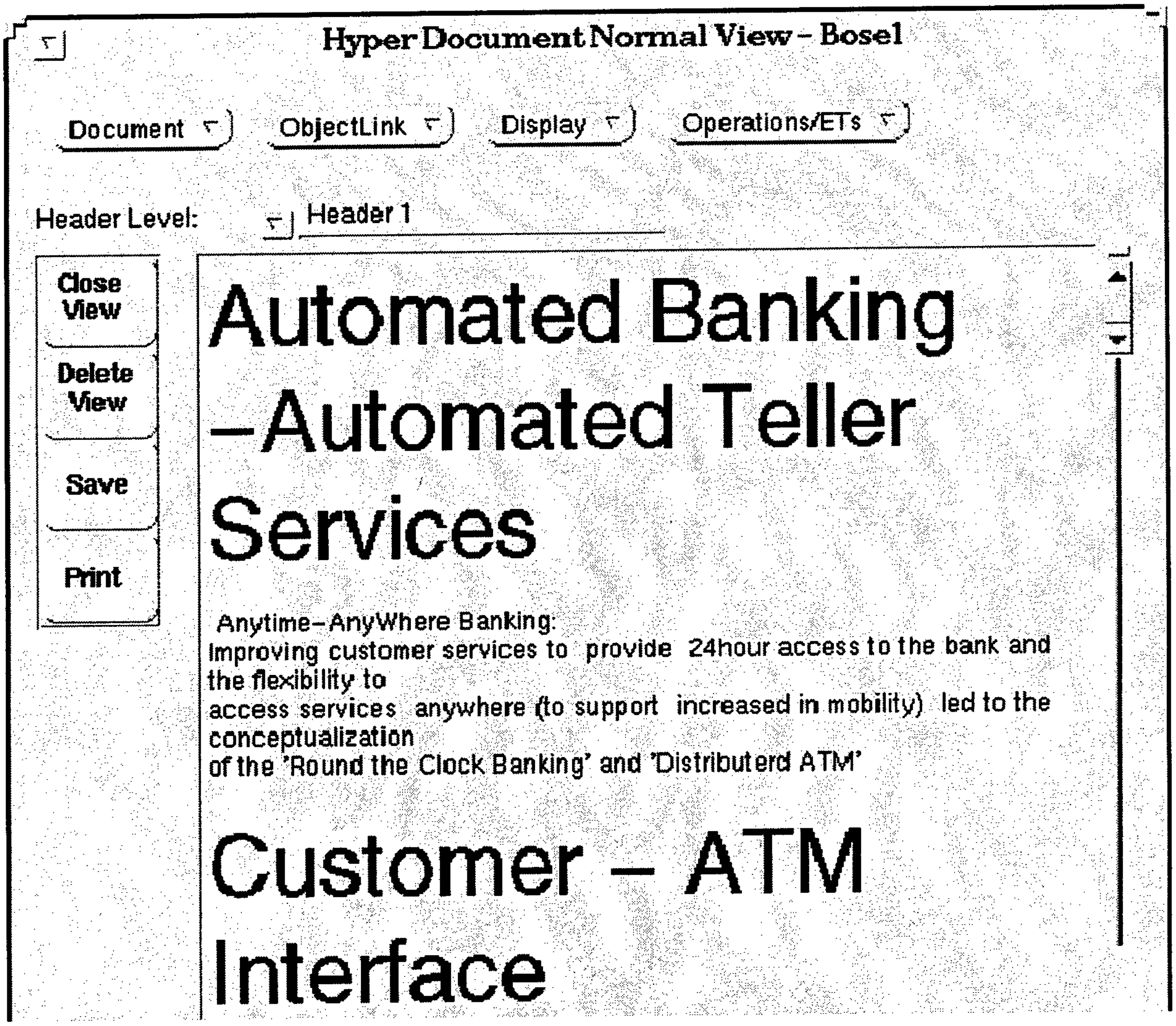
[Selfridge, 1992]. Peter G. Selfridge, "Knowledge Based Software Engineering", IEEE Expert, December 1992.

[Smith, 1991]. D. R. Smith, "KIDS: A Knowledge-Based Software Development System", in Automating Software Design, eds. M. Lowry and R. McCartney, MIT Presss, 1991.

[Winograd, 1996]. T. Winograd, "Bringing Design to Software", Addison Wesley, 1996.

Appendix:

Screen shots of the usage of ADM in the ATM banking example.



Automated Banking –Automated Teller Services

Anytime–AnyWhere Banking:
Improving customer services to provide 24hour access to the bank and the flexibility to
access services anywhere (to support increased in mobility) led to the conceptualization
of the 'Round the Clock Banking' and 'Distributerd ATM'

Customer – ATM Interface

O. Appendix 5. Technology Impact Analysis Tool

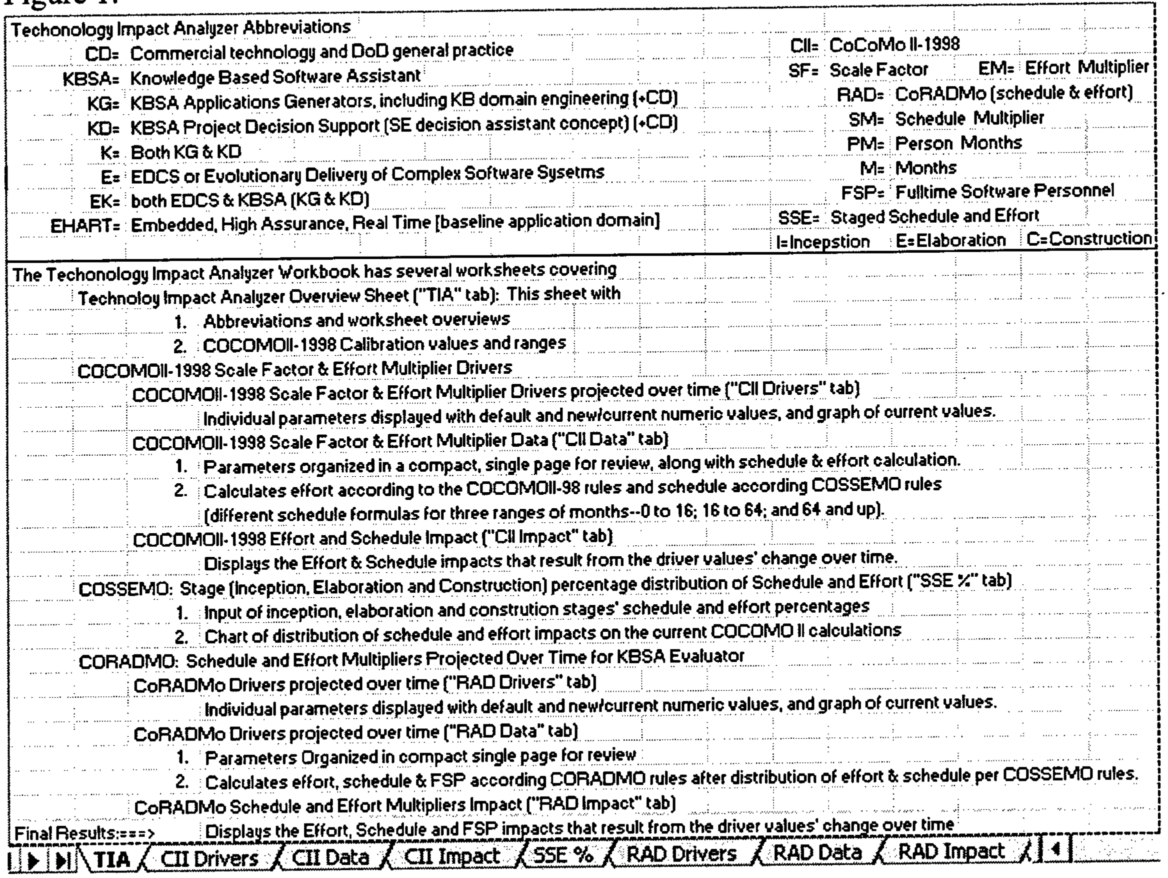
Technology Impact Analysis Tool

A. Winsor Brown

O.1 Tool Overview

A multi-sheet Excel Workbook has been developed to show the impacts of the COCOMO II and CORADMO drivers projected over time and technology-type on a selected domain's typically sized application. This spread sheet model is named "Technology Impact Analyzer" or TIA for short and has the file name TIA.xls. The sheets include an overview and sheets for the COCOMO-II.1998 , COSSEMO and CORADMO drivers, data and their impacts.

The overview sheet includes abbreviations and descriptions of the other sheets on the first page, Figure 1.



Techonology Impact Analyzer Abbreviations

CD=	Commercial technology and DoD general practice	CII=	CoCoMo II-1998
KBSA=	Knowledge Based Software Assistant	SF=	Scale Factor EM= Effort Multiplier
KG=	KBSA Applications Generators, including KB domain engineering (+CD)	RAD=	CoRADMo (schedule & effort)
KD=	KBSA Project Decision Support (SE decision assistant concept) (+CD)	SM=	Schedule Multiplier
K=	Both KG & KD	PM=	Person Months
E=	EDCS or Evolutionary Delivery of Complex Software Sysetms	M=	Months
EK=	both EDCS & KBSA (KG & KD)	FSP=	Fulltime Software Personnel
EHART=	Embedded, High Assurance, Real Time [baseline application domain]	SSE=	Staged Schedule and Effort
			I=Incepstion E=Elaboration C=Construction

The Techonology Impact Analyzer Workbook has several worksheets covering

Technoloy Impact Analyzer Overview Sheet ("TIA" tab): This sheet with

1. Abbreviations and worksheet overviews
2. COCOMOII-1998 Calibration values and ranges

COCOMOII-1998 Scale Factor & Effort Multiplier Drivers

COCOMOII-1998 Scale Factor & Effort Multiplier Drivers projected over time ("CII Drivers" tab)

Individual parameters displayed with default and new/current numeric values, and graph of current values.

COCOMOII-1998 Scale Factor & Effort Multiplier Data ("CII Data" tab)

1. Parameters organized in a compact, single page for review, along with schedule & effort calculation.
2. Calculates effort according to the COCOMOII-98 rules and schedule according COSSEMO rules (different schedule formulas for three ranges of months--0 to 16; 16 to 64; and 64 and up).

COCOMOII-1998 Effort and Schedule Impact ("CII Impact" tab)

Displays the Effort & Schedule impacts that result from the driver values' change over time.

COSSEMO: Stage (Inception, Elaboration and Construction) percentage distribution of Schedule and Effort ("SSE %" tab)

1. Input of inception, elaboration and constrution stages' schedule and effort percentages
2. Chart of distribution of schedule and effort impacts on the current COCOMO II calculations

CORADMO: Schedule and Effort Multipliers Projected Over Time for KBSA Evaluator

CoRADMo Drivers projected over time ("RAD Drivers" tab)

Individual parameters displayed with default and new/current numeric values, and graph of current values.

CoRADMo Drivers projected over time ("RAD Data" tab)

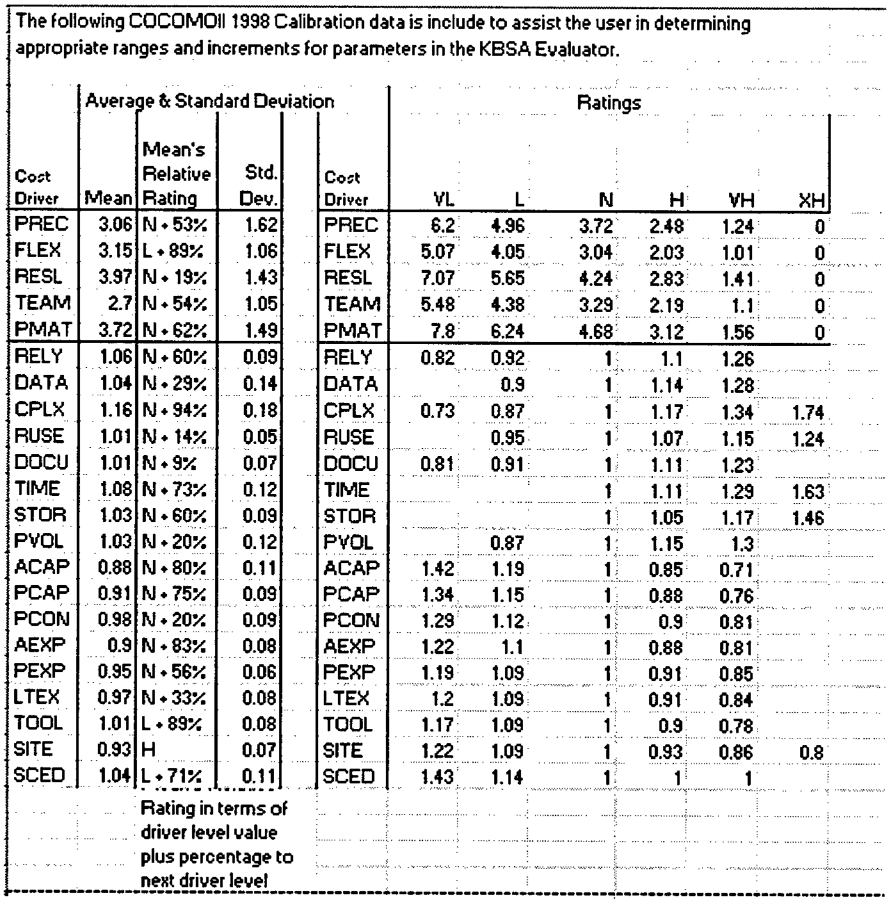
1. Parameters Organized in compact single page for review
2. Calculates effort, schedule & FSP according CORADMO rules after distribution of effort & schedule per COSSEMO rules.

CoRADMo Schedule and Effort Multipliers Impact ("RAD Impact" tab)

Final Results:===> Displays the Effort, Schedule and FSP impacts that result from the driver values' change over time

Figure 1. TIA Abbreviations and Sheet Descriptions

On the second page it has the COCOMO-II.1998 calibration values and ranges for reference, Figure 2.



The following COCOMOII 1998 Calibration data is include to assist the user in determining appropriate ranges and increments for parameters in the KBSA Evaluator.

	Average & Standard Deviation				Ratings					
Cost Driver	Mean	Mean's Relative Rating	Std. Dev.	Cost Driver	VL	L	N	H	VH	XH
PREC	3.06	N + 53%	1.62	PREC	6.2	4.96	3.72	2.48	1.24	0
FLEX	3.15	L + 89%	1.06	FLEX	5.07	4.05	3.04	2.03	1.01	0
RESL	3.97	N + 19%	1.43	RESL	7.07	5.65	4.24	2.83	1.41	0
TEAM	2.7	N + 54%	1.05	TEAM	5.48	4.38	3.29	2.19	1.1	0
PMAT	3.72	N + 62%	1.49	PMAT	7.8	6.24	4.68	3.12	1.56	0
RELY	1.06	N + 60%	0.09	RELY	0.82	0.92	1	1.1	1.26	
DATA	1.04	N + 29%	0.14	DATA		0.9	1	1.14	1.28	
CPLX	1.16	N + 94%	0.18	CPLX	0.73	0.87	1	1.17	1.34	1.74
RUSE	1.01	N + 14%	0.05	RUSE		0.95	1	1.07	1.15	1.24
DOCU	1.01	N + 9%	0.07	DOCU	0.81	0.91	1	1.11	1.23	
TIME	1.08	N + 73%	0.12	TIME			1	1.11	1.29	1.63
STOR	1.03	N + 60%	0.09	STOR			1	1.05	1.17	1.46
PVOL	1.03	N + 20%	0.12	PVOL		0.87	1	1.15	1.3	
ACAP	0.88	N + 80%	0.11	ACAP	1.42	1.19	1	0.85	0.71	
PCAP	0.91	N + 75%	0.09	PCAP	1.34	1.15	1	0.88	0.76	
PCON	0.98	N + 20%	0.09	PCON	1.29	1.12	1	0.9	0.81	
AEXP	0.9	N + 83%	0.08	AEXP	1.22	1.1	1	0.88	0.81	
PEXP	0.95	N + 56%	0.06	PEXP	1.19	1.09	1	0.91	0.85	
LTEX	0.97	N + 33%	0.08	LTEX	1.2	1.09	1	0.91	0.84	
TOOL	1.01	L + 89%	0.08	TOOL	1.17	1.09	1	0.9	0.78	
SITE	0.93	H	0.07	SITE	1.22	1.09	1	0.93	0.86	0.8
SCED	1.04	L + 71%	0.11	SCED	1.43	1.14	1	1	1	
		Rating in terms of driver level value plus percentage to next driver level								

Figure 2. COCOMO-II.1998 Baseline Values

O.1.1 COCOMO-II Drivers, Calculations and Impacts

There are three sheets in this grouping. The first, "CII Drivers", has the current projected scale factors and effort multipliers drivers over time and allows for changing the default values to their new values. The second, "CII Data", aggregates the driver data and does the COCOMO II calculations. The third, "CII Impact", has graphs showing the effort and schedule impact of the COCOMO-II.1998 drivers projected over time.

O.1.2 CoSSEMo Schedule and Effort Percentage Distributions per Stage

This sheet, "SSE %", allows the input of percentage distributions of effort and schedule to the various stages, Inception, Elaboration, and Construction, as required for the COCOMO II Staged

Schedule and Effort Model (COSSEMO). The impact of these distributions on the COCOMO-II.1998 baseline results is shown in the chart at the end of the worksheet.

O.1.3 CORADMO Drivers, Calculations and Impacts

Like the COCOMO-II.1998 sheets, there are three sheets in this grouping. The first, "RAD Drivers", shows the new or default projected drivers over time. The second, "RAD Data", aggregates the driver data and does the CoRADMo calculations. The third, "RAD Impact", has graphs showing the resulting impacts of the CoRADMo drivers projected over time when applied to the corresponding COCOMO-II.1998 results with the COCOMO drivers projected over time. At the end of the page of the "RAD Data" sheet are the summary calculations for totals of schedule and effort across stages allowing comparison with the results of COCOMO-II.1998.

O.1.4 Technical Impact Final Results

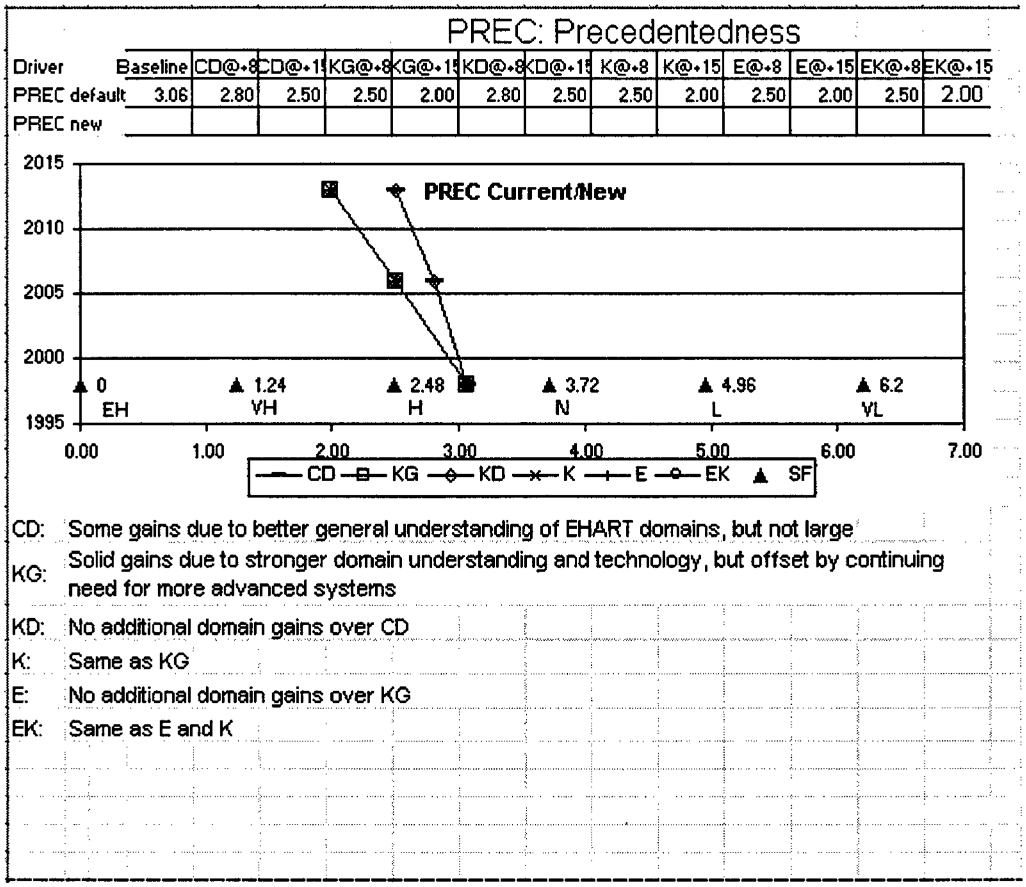
At the end of the "RAD Impact" worksheet, following the nine RAD impacts by stage charts, are the summary charts for effort and schedule by technology over time that result from the COCOMO-II.1998 and CORADMO driver changes over time. The effort and schedule results are generated by adding the effort or schedule, respectively, for either all three stages or just for the Elaboration and Construction stages.

O.2 CoCoMo II Drivers, Calculation and Display of Impacts

The three sheets in this grouping show the driver data, COCOMO-II.1998 calculations, and the impacts of the projected drivers over time and technology.

O.2.1 CII Drivers – Display, Modification and Rationale

"CII Drivers" shows all of our assessed values for each of the scale factor or effort multiplier drivers, projected over time and technology, and our rationale. Each page of this worksheet has the current projected COCOMO-II.1998 drivers, both scale factors and effort multipliers, over time and allows for changing the default values to their new values. The rationales for the default settings of the drivers are included; they should be modified when “new” values are provided. Figure 3 shows the scale factor PREC’s information.



PREC: Precedentedness

Driver	Baseline	CD@+8	CD@+1	KG@+8	KG@+1	KD@+8	KD@+1	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
PREC default	3.06	2.80	2.50	2.50	2.00	2.80	2.50	2.50	2.00	2.50	2.00	2.50	2.00
PREC new													

CD: Some gains due to better general understanding of EHART domains, but not large

KG: Solid gains due to stronger domain understanding and technology, but offset by continuing need for more advanced systems

KD: No additional domain gains over CD

K: Same as KG

E: No additional domain gains over KG

EK: Same as E and K

Figure 3. PREC's Driver Entry, Modification and Display

The default and current values of the driver, projected over time and technology, are shown in a small table above the chart of the current values. The last row of this table accepts the input of new values of the driver, projected over time and technology. The chart below the table shows the driver's current values over time for each technology combination. The data points on this graph change when new values are entered.

Since each value of a driver should have a rationale, the rationales for the default values (our assessed values) are shown below the chart. The area below the rationales for the default values allows the input of additional or modified rationales.

O.2.2 CoCoMo II Calculations

"CII Data" has the current assessed COCOMO-II.1998 drivers, both scale factors and effort multipliers, organized in a compact, single page sheet along with the calculations of the COCOMO II effort and schedule. The calculations use the COCOMO-II.1998 model equations for effort and schedule, and then applies the COSSEMO equations for schedule (different schedule formulas for three ranges of person-months of effort: 0 to 16; 16 to 64; and 64 and up). Each column of the table performs the full set of COCOMO-II calculations for a particular year and technology-type combination. The worksheet is shown in Figure 4.

Tech Impact Analyzer: COCOMOII-1998 Scale Factors & Effort Multipliers -- Now, +8 & +15 years													
	COCOMO II data (from COCOMO II Drivers) and calculations												
Factors into future													
	Baseline	CD	CD	KG	KG	KD	KD	K	K	EDCS	EDCS	EK	EK
Cost-Driver	(now)	8Yr	15Yr	8Yr	15Yr	8Yr	15Yr	8Yr	15Yr	8Yr	15Yr	8Yr	15Yr
	1998	2006	2013	2006	2013	2006	2013	2006	2013	2006	2013	2006	2013
PREC	3.06	2.8	2.5	2.5	2	2.8	2.5	2.5	2	2.5	2	2.5	2
FLEX	3.15	2.8	2.5	2.5	2	2.6	2.3	2.4	1.8	2.4	1.8	2.2	1.5
RESL	3.97	3.5	3	3.2	2.5	3.2	2.5	3	2.2	3	2.2	2.7	1.7
TEAM	2.7	2.4	2	2.4	2	2	1.4	2	1.4	2.1	1.6	1.8	1.1
PMAT	3.72	3	2.2	3	2.2	2.8	1.8	2.8	1.8	3	2.2	2.8	1.8
Σ	16.6	14.5	12.2	13.6	10.7	13.4	10.5	12.7	9.2	13	9.8	12	8.1
B	1.076	1.055	1.032	1.046	1.017	1.044	1.015	1.037	1.002	1.04	1.01	1.03	0.991
RELY	1.16	1.14	1.12	1.14	1.12	1.14	1.12	1.14	1.12	1.1	1.06	1.1	1.06
DATA	1.04	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98
CPLX	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16	1.16
RUSE	1.01	1.05	1.03	1.1	1.08	1.05	1.03	1.1	1.08	1.1	1.08	1.1	1.08
DOCU	1.01	0.97	0.95	0.93	0.91	0.95	0.93	0.92	0.89	0.92	0.89	0.92	0.89
TIME	1.2	1.12	1.08	1.12	1.08	1.12	1.08	1.12	1.08	1.08	1.04	1.08	1.04
STOR	1.08	1.04	1.02	1.04	1.02	1.04	1.02	1.04	1.02	1.04	1.02	1.04	1.02
PVOL	1.03	1.01	0.98	1.01	0.98	1.01	0.98	1.01	0.98	0.98	0.94	0.98	0.94
ACAP	0.88	0.88	0.88	0.88	0.88	0.86	0.82	0.86	0.82	0.88	0.88	0.86	0.82
PCAP	0.91	0.91	0.91	0.91	0.91	0.89	0.85	0.89	0.85	0.91	0.91	0.89	0.85
PCON	0.98	0.98	0.98	0.98	0.98	0.96	0.93	0.96	0.93	0.96	0.93	0.95	0.91
AEXP	0.9	0.89	0.87	0.87	0.84	0.89	0.87	0.87	0.84	0.87	0.84	0.87	0.84
PEXP	0.95	0.94	0.92	0.94	0.92	0.94	0.92	0.94	0.92	0.94	0.92	0.94	0.92
LTEX	0.97	0.95	0.92	0.93	0.89	0.95	0.92	0.93	0.89	0.93	0.89	0.93	0.89
TOOL	1.01	0.98	0.94	0.98	0.94	0.96	0.9	0.96	0.9	0.94	0.88	0.94	0.86
SITE	0.93	0.91	0.88	0.91	0.88	0.91	0.88	0.91	0.88	0.88	0.84	0.88	0.84
SCED	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04
Π	1.21	0.93	0.67	0.89	0.63	0.83	0.52	0.81	0.49	0.72	0.45	0.68	0.38
ΠnoSCED	1.17	0.89	0.64	0.86	0.60	0.80	0.50	0.78	0.47	0.69	0.44	0.66	0.36
SCED%	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71	95.71
SIZE Orig.	100	60	30	40	15	60	30	40	15	35	12	30	10
SIZE	100	60	30	40	15	60	30	40	15	35	12	30	10
CII_PM	505.48	204.55	65.76	124.00	28.97	175.65	48.05	108.83	21.51	85.66	16.37	66.71	10.90
CII_PMnoS	486.04	196.69	63.23	119.23	27.85	168.90	46.20	104.65	20.68	82.37	15.74	64.15	10.48
CII_PM Orig	505.48	204.55	65.76	124.00	28.97	175.65	48.05	108.83	21.51	85.66	16.37	66.71	10.90
CII_M Orig.	24.38	17.96	12.41	15.26	9.57	16.95	11.14	14.54	8.67	13.55	8.02	12.45	7.05
CII_Mof64	12.92	12.70	12.46	12.60	12.30	12.58	12.28	12.51	12.15	12.54	12.21	12.44	12.04
SSEMo_M	24.38	17.96	12.32	15.26	6.05	16.95	9.21	14.54	4.80	13.55	3.97	12.45	3.24
SSEMo_M	24.38	17.96	12.32	15.26	6.05	16.95	9.21	14.54	4.80	13.55	3.97	12.45	3.24

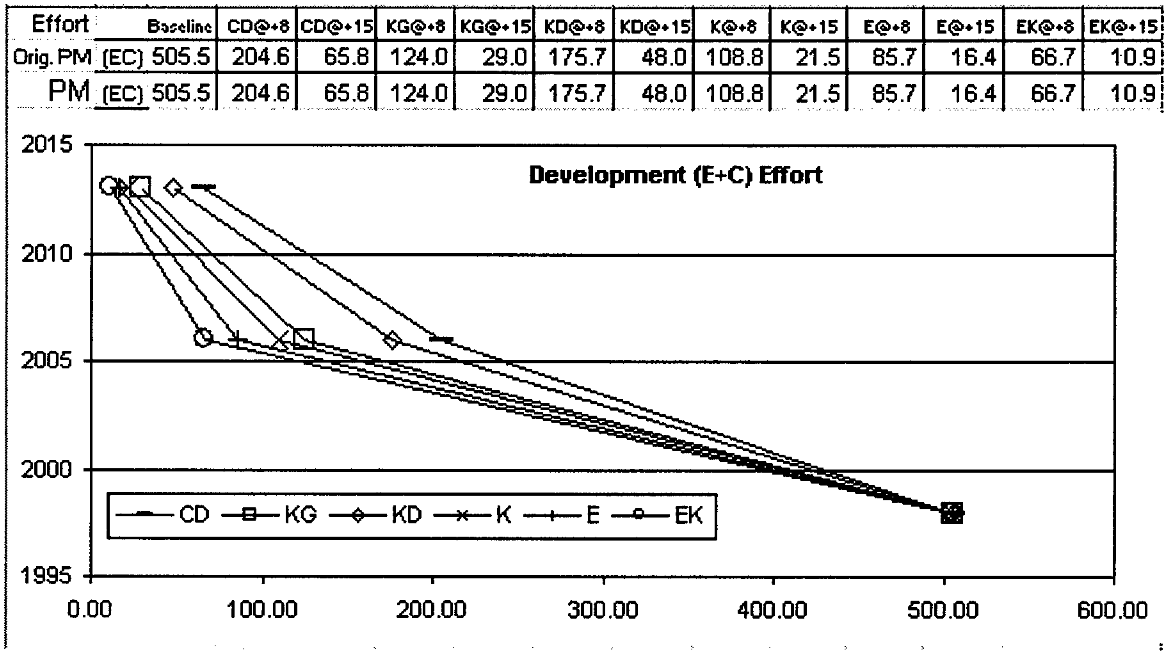
Figure 4. CII Data Worksheet

In the sheet the following non-driver abbreviations, in their order of appearance, are used.

Abbreviation	Meaning	Abbreviation	Meaning
Σ	Sum of the scale factors	CII_PM	COCOMO-II.1998 effort
B	The exponent for effort calculation	CII_PMnoSCED	COCOMO-II.1998 effort without SCED
Π	The product of the effort multipliers	CII_PM Orig.	COCOMO-II.1998 effort using default drivers
ΠnoSCED	Effort multiplier's product without SCED	CII_M Orig.	COCOMO-II.1998 schedule [Original]
SCED%	The schedule compression percentage.	CII_Mof64	COCOMO-II.1998 schedule at 64 PM
SIZE Orig.	The original (default) SIZE value	SSEMo_M	COSSEMO schedule in months
SIZE	The current SIZE value	SSEMo_M Orig.	COSSEMO schedule using default drivers

O.2.3 COCOMOII-1998 Effort and Schedule Impacts

This work sheet displays the Effort & Schedule impacts that result from the driver values' change over time and technology. The impacts are shown in both tabular and chart form, with the chart always reflecting the "current" values of the drivers. An example is shown in Figure 5.



Effort		Baseline	CD@+8	CD@+15	KG@+8	KG@+15	KD@+8	KD@+15	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
Orig. PM	(EC)	505.5	204.6	65.8	124.0	29.0	175.7	48.0	108.8	21.5	85.7	16.4	66.7	10.9
PM	(EC)	505.5	204.6	65.8	124.0	29.0	175.7	48.0	108.8	21.5	85.7	16.4	66.7	10.9

Figure 5. COCOMO-II.1998 based Development Effort Impact Example

The table above these charts shows the calculated results based on the default driver's values and the updated values based on the "new" values of the drivers. Where there are multiple calculations that might provide useful information, those intermediate results are also shown, as in Figure 6.

Schedule	Baseline	CD@+8	CD@+15	KG@+8	KG@+15	KD@+8	KD@+15	K@+8	K@+15	E@+8	E@+15	EK@+8	EK@+15
CII's M Orig	24.4	18.0	12.4	15.3	9.6	16.9	11.1	14.5	8.7	13.5	8.0	12.4	7.0
SSEM-Orig (EC)	24.4	18.0	12.3	15.3	6.1	16.9	9.2	14.5	4.8	13.5	4.0	12.4	3.2
SSE M (EC)	24.4	18.0	12.3	15.3	6.1	16.9	9.2	14.5	4.8	13.5	4.0	12.4	3.2

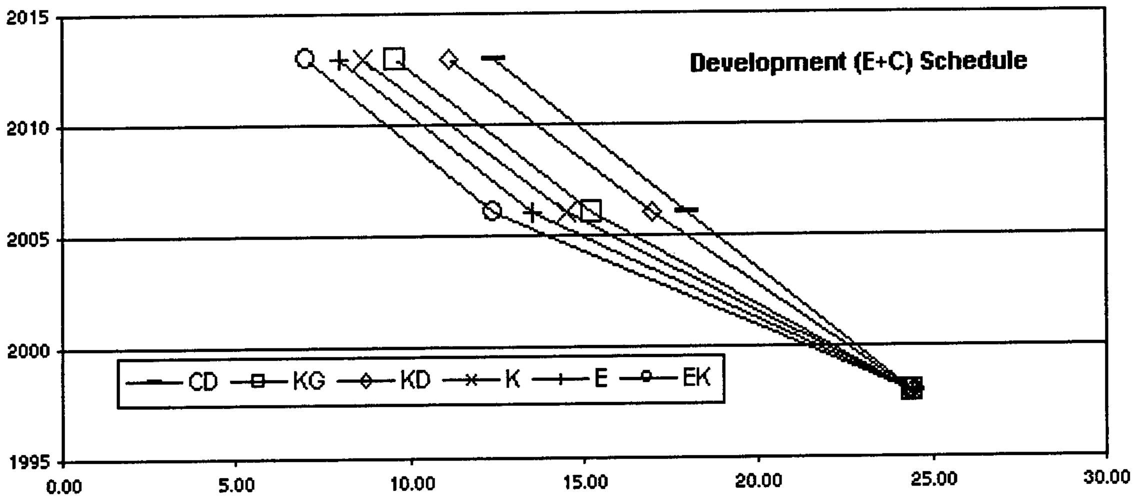


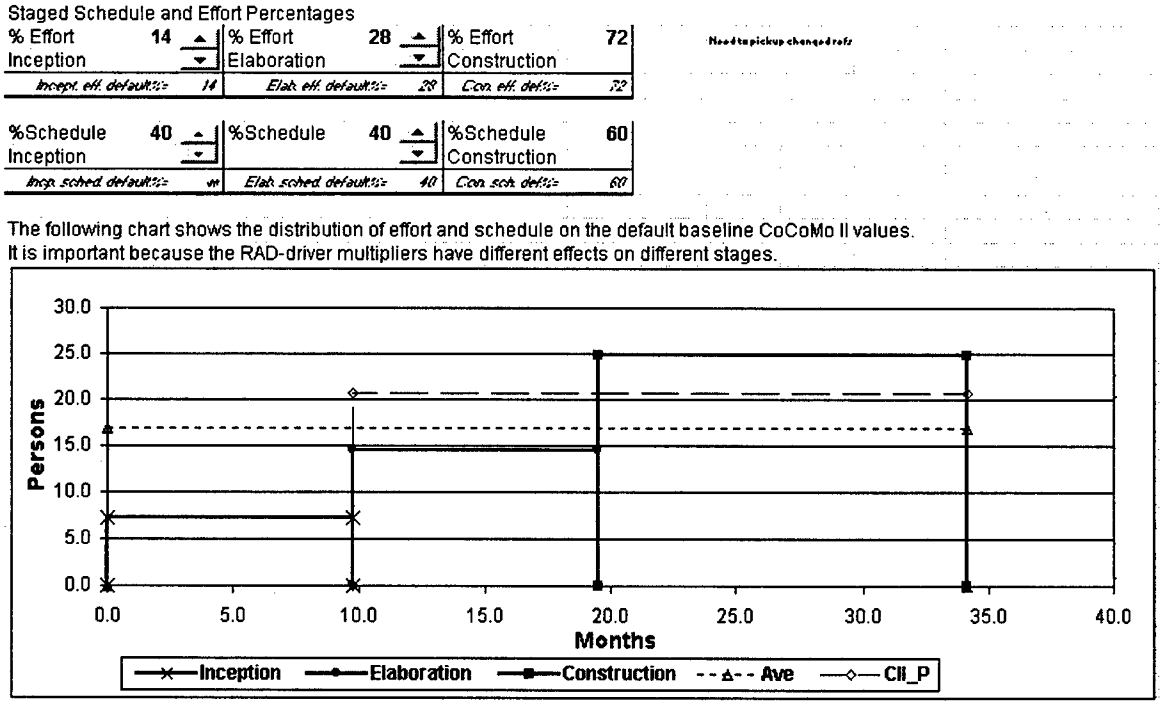
Figure 6. COCOMO-II.1998 based Development Schedule Impact Example

Here, both the original COCOMO-II.1998 set of calculations and the COSSEMO-based set of calculations are shown. Again, the final row's values will contain the results based on the "current" driver values, and thus may have changes anytime there is input in the "new" row of the drivers.

O.3 COSSEMO Distribution of Schedule and Effort per Stage

There are two parts to this worksheet: 1) Input of inception, elaboration and construction stages' schedule and effort percentages; and 2) Chart of distribution of schedule and effort impacts on the current COCOMO II calculations.

Input of schedule and effort percentage distributions per stage, Inception, Elaboration, and Construction, is required for the COCOMO II Staged Schedule and Effort Model (COSSEMO). To help visualize these distributions, their impact on the COCOMO-II.1998 100K EHART baseline is displayed in the chart at the end of the worksheet. Figure 8 shows the entire content of this worksheet.



Staged Schedule and Effort Percentages

% Effort Inception	14	% Effort Elaboration	28	% Effort Construction	72
Incept. eff. default%=	*14*	*Elab. eff. default%=*	*28*	*Con. eff. def%=*	*72*

Need to pickup changed refs

%Schedule Inception	40	%Schedule	40	%Schedule Construction	60
Incp. sched default%=	*40*	*Elab. sched default%=*	*40*	*Con. sch. def%=*	*60*

The following chart shows the distribution of effort and schedule on the default baseline CoCoMo II values.
It is important because the RAD-driver multipliers have different effects on different stages.

Figure 7. Staged Schedule and Effort Distribution.

The values of the Inception and Elaboration percentages for schedule and effort are adjusted by clicking on the up/down arrows (spinners) shown to the right of their values. The current values are displayed in bold, along with the corresponding calculated values for the Construction stage. The default values for all the percentages are shown in italics.

The chart that follows the input area shows the impact of the distributions on the calculated baseline results. Since COCOMO-II.1998 only calculates the effort and schedule for the Elaboration plus Construction stage, the corresponding Fulltime Software Personnel (FSP; AKA "Persons"), labeled CII_P, is shown only for that duration.

O.4 CORADMO Drivers, Data and Impacts

The three sheets in this grouping show the driver data, CORADMO calculations, and the impacts of the projected drivers over time and technology.

O.4.1 CORADMO Drivers - Display, Modification and Rationale

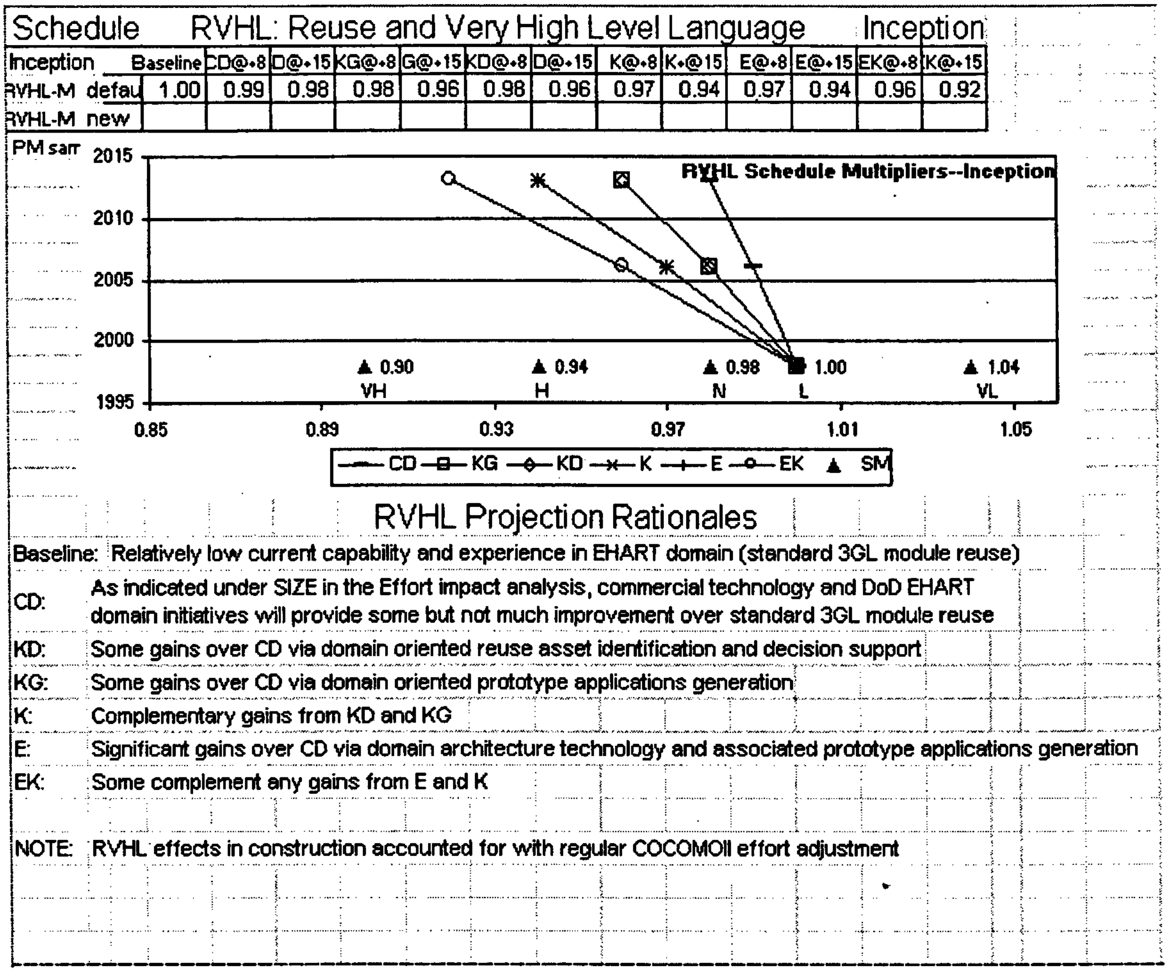
"RAD Drivers" has our assessed values for each of the relevant CORADMO schedule and effort multipliers projected over time and our rationale. It also allows the input of new values and additional or modified rationales. A graph of the current values of each driver projected over time and technology is included; the data points on this graph change when new values are entered.

There are five CORADMO drivers (RVHL, DPRS, CLAB, RESL and PPOS):

1. RVHL: Reuse and Very High Level Language
2. DPRS: Development Process Reengineering & Streamlining
3. CLAB: Collaboration Efficiency
4. RESL: Architecture/Risk Resolution
5. PPOS: Prepositioning Assets

With five CORADMO drivers, three stages (Inception, Elaboration and Construction), and two multipliers (effort and schedule; two of the three variables in "Person Months = Persons * Months", or PM=P*M, equation), there are 30 different driver possibilities. How ever, there are several situations with reduce the actual numbers of drivers in the Technical Impact Analyzer. The number of persons is held constant for RVHL, DPRS, CLAB and RESL, and therefore the drivers for effort and schedule have the same value. The impact of RVHL on construction is handled by/in the reuse model of regular COCOMO-II.1998. The impact of DPRS is assumed to be the same for Elaboration and Construction. And, while PPOS has different multipliers for effort and schedule, the same values are used for all three stages. Thus the number of drivers is reduced to ten from thirty, although an eleventh chart is included in this worksheet to show the effect of the PPOS drivers on the number of personnel.

"RAD Drivers" shows all of our assessed values for the significant CORADMO drivers , projected over time and technology, and our rationale. Each page of this worksheet, with the exception of the last, has the current projected CORADMO drivers, over time and allows for changing the default values to their new values (the exception is for PPOS's FSP driver which is a derived value). The rationales for the default settings of the drivers are included; they should be modified when "new" values are provided. Figure 9 shows RVHL's Inception-Schedule Multiplier Driver Information.



Schedule RVHL: Reuse and Very High Level Language Inception

Inception	Baseline	CD@+8	D@+15	KG@+8	G@+15	KD@+8	D@+15	K@+8	K+@15	E@+8	E@+15	EK@+8	K@+15
RVHL-M defau	1.00	0.99	0.98	0.98	0.96	0.98	0.96	0.97	0.94	0.97	0.94	0.96	0.92
RVHL-M new													

RVHL Projection Rationales

Baseline: Relatively low current capability and experience in EHART domain (standard 3GL module reuse)

CD: As indicated under SIZE in the Effort impact analysis, commercial technology and DoD EHART domain initiatives will provide some but not much improvement over standard 3GL module reuse

KD: Some gains over CD via domain oriented reuse asset identification and decision support

KG: Some gains over CD via domain oriented prototype applications generation

K: Complementary gains from KD and KG

E: Significant gains over CD via domain architecture technology and associated prototype applications generation

EK: Some complement any gains from E and K

NOTE: RVHL effects in construction accounted for with regular COCOMOII effort adjustment

Figure 8. RVHL's Inception Stage Schedule Multiplier Driver Information

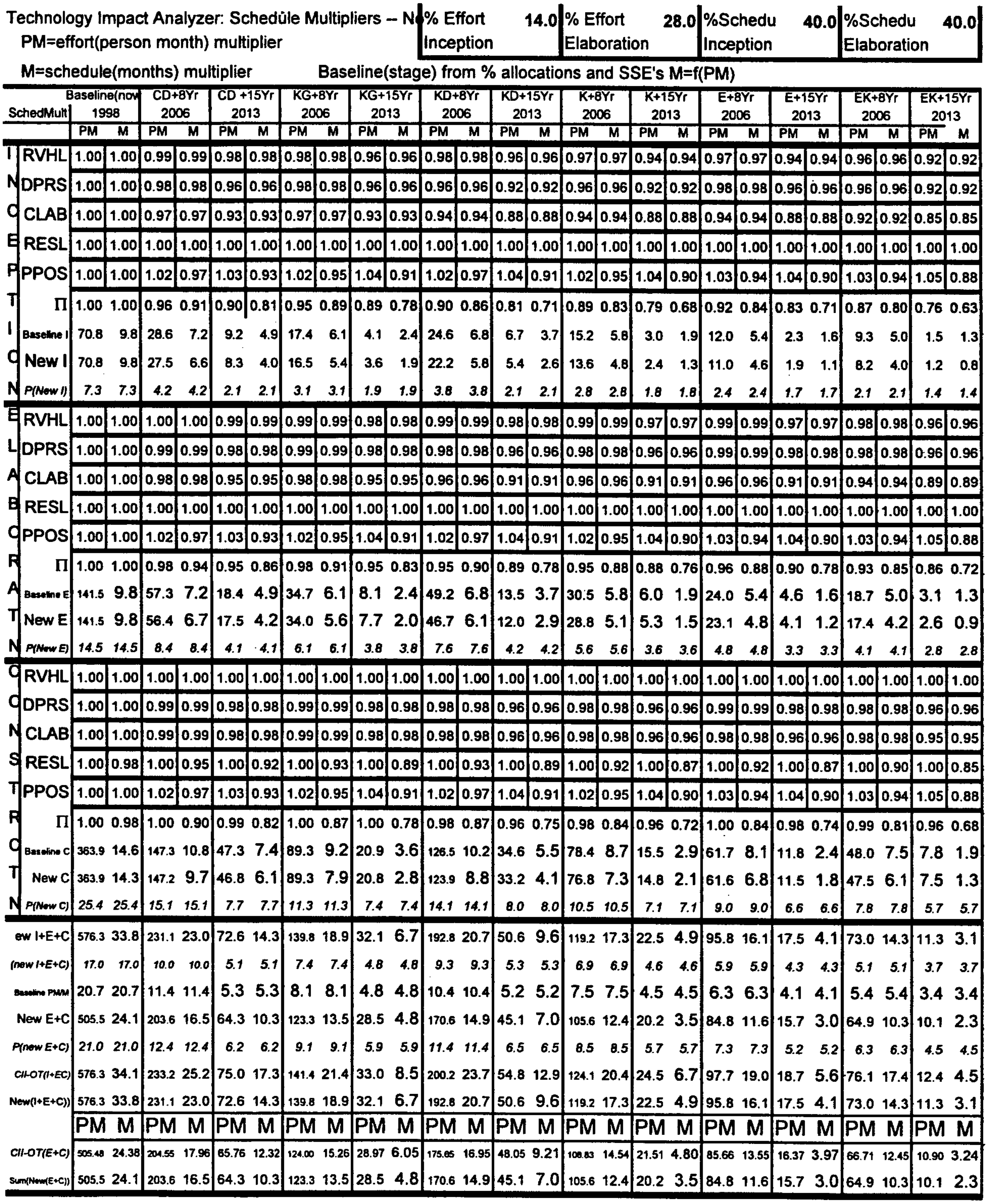
The default and current values of the driver, projected over time and technology, are shown in a small table above the chart of the current values. The last row of this table accepts the input of new values of the driver, projected over time and technology. The chart below the table shows the driver's values over time for each technology combination.

Since each value of a driver should have a rationale, the rationales for the default values (our assessed values) are shown below the chart.

O.4.2 CORADMO Calculations

"RAD Data" aggregates the CORADMO drivers, both schedule and effort multipliers. They are organized in a compact, single page sheet along with the calculations of the CORADMO effort and schedule. The calculations use the CORADMO model equations to distribute schedule and effort based on the selected percentage allocations and the schedule or effort multiplier driver ratings. Each Person-Month (PM) & Month (M) pair of data columns of the table performs the full set of CORADMO calculations, including the derivable Personnel (P) values, for a particular year and technology-type combination.

At the end of the page of the "RAD Data" worksheet are the summary calculations for totals of schedule and effort across stages allowing comparison with the results of COCOMO-II.1998.



Technology Impact Analyzer: Schedůle Multipliers -- N

PM=effort(person month) multiplier

M=schedule(months) multiplier Baseline(stage) from % allocations and SSE's M=f(PM)

% Effort Inception	14.0	% Effort Elaboration	28.0	%Schedu Inception	40.0	%Schedu Elaboration	40.0

		Baseline(now		CD+8Yr		CD +15Yr		KG+8Yr		KG+15Yr		KD+8Yr		KD+15Yr		K+8Yr		K+15Yr		E+8Yr		E+15Yr		EK+8Yr		EK+15Yr	
	SchedMult	1998		2006		2013		2006		2013		2006		2013		2006		2013		2006		2013		2006		2013	
		PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M
I	RVHL	1.00	1.00	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.97	0.97	0.94	0.94	0.97	0.97	0.94	0.94	0.96	0.96	0.92	0.92
N	DPRS	1.00	1.00	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.96	0.96	0.92	0.92	0.96	0.96	0.92	0.92	0.98	0.98	0.96	0.96	0.96	0.96	0.92	0.92
C	CLAB	1.00	1.00	0.97	0.97	0.93	0.93	0.97	0.97	0.93	0.93	0.94	0.94	0.88	0.88	0.94	0.94	0.88	0.88	0.94	0.94	0.88	0.88	0.92	0.92	0.85	0.85
E	RESL	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
P	PPOS	1.00	1.00	1.02	0.97	1.03	0.93	1.02	0.95	1.04	0.91	1.02	0.97	1.04	0.91	1.02	0.95	1.04	0.90	1.03	0.94	1.04	0.90	1.03	0.94	1.05	0.88
T	Π	1.00	1.00	0.96	0.91	0.90	0.81	0.95	0.89	0.89	0.78	0.90	0.86	0.81	0.71	0.89	0.83	0.79	0.68	0.92	0.84	0.83	0.71	0.87	0.80	0.76	0.63
I	Baseline I	70.8	9.8	28.6	7.2	9.2	4.9	17.4	6.1	4.1	2.4	24.6	6.8	6.7	3.7	15.2	5.8	3.0	1.9	12.0	5.4	2.3	1.6	9.3	5.0	1.5	1.3
O	New I	70.8	9.8	27.5	6.6	8.3	4.0	16.5	5.4	3.6	1.9	22.2	5.8	5.4	2.6	13.6	4.8	2.4	1.3	11.0	4.6	1.9	1.1	8.2	4.0	1.2	0.8
N	*P(New I)*	*7.3*	*7.3*	*4.2*	*4.2*	*2.1*	*2.1*	*3.1*	*3.1*	*1.9*	*1.9*	*3.8*	*3.8*	*2.1*	*2.1*	*2.8*	*2.8*	*1.8*	*1.8*	*2.4*	*2.4*	*1.7*	*1.7*	*2.1*	*2.1*	*1.4*	*1.4*
E	RVHL	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.99	0.98	0.98	0.99	0.99	0.98	0.98	0.99	0.99	0.97	0.97	0.99	0.99	0.97	0.97	0.98	0.98	0.96	0.96
L	DPRS	1.00	1.00	0.99	0.99	0.98	0.98	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96
A	CLAB	1.00	1.00	0.98	0.98	0.95	0.95	0.98	0.98	0.95	0.95	0.96	0.96	0.91	0.91	0.96	0.96	0.91	0.91	0.96	0.96	0.91	0.91	0.94	0.94	0.89	0.89
B	RESL	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
O	PPOS	1.00	1.00	1.02	0.97	1.03	0.93	1.02	0.95	1.04	0.91	1.02	0.97	1.04	0.91	1.02	0.95	1.04	0.90	1.03	0.94	1.04	0.90	1.03	0.94	1.05	0.88
R	Π	1.00	1.00	0.98	0.94	0.95	0.86	0.98	0.91	0.95	0.83	0.95	0.90	0.89	0.78	0.95	0.88	0.88	0.76	0.96	0.88	0.90	0.78	0.93	0.85	0.86	0.72
A	Baseline E	141.5	9.8	57.3	7.2	18.4	4.9	34.7	6.1	8.1	2.4	49.2	6.8	13.5	3.7	30.5	5.8	6.0	1.9	24.0	5.4	4.6	1.6	18.7	5.0	3.1	1.3
T	New E	141.5	9.8	56.4	6.7	17.5	4.2	34.0	5.6	7.7	2.0	46.7	6.1	12.0	2.9	28.8	5.1	5.3	1.5	23.1	4.8	4.1	1.2	17.4	4.2	2.6	0.9
N	*P(New E)*	*14.5*	*14.5*	*8.4*	*8.4*	*4.1*	*4.1*	*6.1*	*6.1*	*3.8*	*3.8*	*7.6*	*7.6*	*4.2*	*4.2*	*5.6*	*5.6*	*3.6*	*3.6*	*4.8*	*4.8*	*3.3*	*3.3*	*4.1*	*4.1*	*2.8*	*2.8*
C	RVHL	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
O	DPRS	1.00	1.00	0.99	0.99	0.98	0.98	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96
N	CLAB	1.00	1.00	0.99	0.99	0.98	0.98	0.99	0.99	0.98	0.98	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.98	0.98	0.96	0.96	0.98	0.98	0.95	0.95
S	RESL	1.00	0.98	1.00	0.95	1.00	0.92	1.00	0.93	1.00	0.89	1.00	0.93	1.00	0.89	1.00	0.92	1.00	0.87	1.00	0.92	1.00	0.87	1.00	0.90	1.00	0.85
T	PPOS	1.00	1.00	1.02	0.97	1.03	0.93	1.02	0.95	1.04	0.91	1.02	0.97	1.04	0.91	1.02	0.95	1.04	0.90	1.03	0.94	1.04	0.90	1.03	0.94	1.05	0.88
R	Π	1.00	0.98	1.00	0.90	0.99	0.82	1.00	0.87	1.00	0.78	0.98	0.87	0.96	0.75	0.98	0.84	0.96	0.72	1.00	0.84	0.98	0.74	0.99	0.81	0.96	0.68
C	Baseline C	363.9	14.6	147.3	10.8	47.3	7.4	89.3	9.2	20.9	3.6	126.5	10.2	34.6	5.5	78.4	8.7	15.5	2.9	61.7	8.1	11.8	2.4	48.0	7.5	7.8	1.9
T	New C	363.9	14.3	147.2	9.7	46.8	6.1	89.3	7.9	20.8	2.8	123.9	8.8	33.2	4.1	76.8	7.3	14.8	2.1	61.6	6.8	11.5	1.8	47.5	6.1	7.5	1.3
N	*P(New C)*	*25.4*	*25.4*	*15.1*	*15.1*	*7.7*	*7.7*	*11.3*	*11.3*	*7.4*	*7.4*	*14.1*	*14.1*	*8.0*	*8.0*	*10.5*	*10.5*	*7.1*	*7.1*	*9.0*	*9.0*	*6.6*	*6.6*	*7.8*	*7.8*	*5.7*	*5.7*
	ew I+E+C	576.3	33.8	231.1	23.0	72.6	14.3	139.8	18.9	32.1	6.7	192.8	20.7	50.6	9.6	119.2	17.3	22.5	4.9	95.8	16.1	17.5	4.1	73.0	14.3	11.3	3.1
	(new I+E+C)	*17.0*	*17.0*	*10.0*	*10.0*	*5.1*	*5.1*	*7.4*	*7.4*	*4.8*	*4.8*	*9.3*	*9.3*	*5.3*	*5.3*	*6.9*	*6.9*	*4.6*	*4.6*	*5.9*	*5.9*	*4.3*	*4.3*	*5.1*	*5.1*	*3.7*	*3.7*
	Baseline PM/M	20.7	20.7	11.4	11.4	5.3	5.3	8.1	8.1	4.8	4.8	10.4	10.4	5.2	5.2	7.5	7.5	4.5	4.5	6.3	6.3	4.1	4.1	5.4	5.4	3.4	3.4
	New E+C	505.5	24.1	203.6	16.5	64.3	10.3	123.3	13.5	28.5	4.8	170.6	14.9	45.1	7.0	105.6	12.4	20.2	3.5	84.8	11.6	15.7	3.0	64.9	10.3	10.1	2.3
	P(new E+C)	*21.0*	*21.0*	*12.4*	*12.4*	*6.2*	*6.2*	*9.1*	*9.1*	*5.9*	*5.9*	*11.4*	*11.4*	*6.5*	*6.5*	*8.5*	*8.5*	*5.7*	*5.7*	*7.3*	*7.3*	*5.2*	*5.2*	*6.3*	*6.3*	*4.5*	*4.5*
	CII-OT(I+EC)	576.3	34.1	233.2	25.2	75.0	17.3	141.4	21.4	33.0	8.5	200.2	23.7	54.8	12.9	124.1	20.4	24.5	6.7	97.7	19.0	18.7	5.6	76.1	17.4	12.4	4.5
	New(I+E+C))	576.3	33.8	231.1	23.0	72.6	14.3	139.8	18.9	32.1	6.7	192.8	20.7	50.6	9.6	119.2	17.3	22.5	4.9	95.8	16.1	17.5	4.1	73.0	14.3	11.3	3.1
		PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M	PM	M
	CII-OT(E+C)	505.48	24.38	204.55	17.96	65.76	12.32	124.00	15.26	28.97	6.05	175.65	16.95	48.05	9.21	108.83	14.54	21.51	4.80	85.66	13.55	16.37	3.97	66.71	12.45	10.90	3.24
	Sum(New(E+C))	505.5	24.1	203.6	16.5	64.3	10.3	123.3	13.5	28.5	4.8	170.6	14.9	45.1	7.0	105.6	12.4	20.2	3.5	84.8	11.6	15.7	3.0	64.9	10.3	10.1	2.3

Figure 9. RAD Data Worksheet

In the sheet, the following additional non-driver abbreviations, in their order of appearance, are used.

Abbreviation	Meaning
Π	The product of the five RAD drivers above.
Baseline I (or E or C)	The COCOMO-II-1998 calculated value after applying the Staged Schedule and Effort percentage distribution for Inception, Elaboration or Construction, respectively.
New I (or E or C)	The new value, i.e. after applying the RAD drivers, for Inception, Elaboration or Construction, respectively.
P(New I) (or E or C)	The number of Fulltime Software Personnel corresponding to the new values for Inception, Elaboration or Construction, respectively.
New I+E+C	The new value (PM or M, depending on the column), i.e. after applying the RAD drivers, combined for Inception, Elaboration and Construction
P(new I+E+C)	The number of Fulltime Software Personnel corresponding to the new values for Inception, Elaboration or Construction.
Baseline PM/M	Provided for reference purposes, it shows the "baseline" (the sum for all stages of the COCOMO-II-1998 calculated value after applying the Staged Schedule and Effort percentage distributions) Fulltime Software Personnel.
New E+C	The new value (PM or M, depending on the column) for the combination of the Elaboration and Construction stages. This corresponds to the stages over which COCOMO-II.1998 calculations apply.
P(new E+C)	The new Fulltime Software Personnel (FSP and P) calculations for the combination of the Elaboration and Construction stages.
CII-OT(I+EC)	COCOMO-II.1998 values, applying the projected drivers over time and technology, for the same stages as the summed CORADMO calculations; i.e. the sum for all stages of the COCOMO-II-1998 calculated values after applying the Staged Schedule and Effort percentage distributions. Since COCOMO-II.1998 does NOT include an Inception stage, the additional percentage from the SSE distribution is used. Provided for reference purposes, it shows a "baseline" prior to apply the RAD drivers.
Sum(New(I+E+C))	Repeated values, equivalent to New I+E+C, for the new value (PM or M, depending on the column), i.e. after applying the RAD drivers, combined for Inception, Elaboration and Construction.
CII-OT(E+C)	COCOMO-II.1998 values, applying the projected drivers over time and technology, for the its covered stages; i.e. for the Elaboration and Construction. Provided for reference purposes, it shows a "baseline" prior to apply the RAD drivers.
Sum(New(E+C))	The new value (PM or M, depending on the column) for the combination of the Elaboration and Construction stages. This corresponds to the stages over which COCOMO-II.1998 calculations apply. Provided to ease comparison with row above.

O.4.3 CoRADMo

O.4.4 Effort and Schedule Impacts

"RAD Impact" has graphs showing the effort, schedule and Full-time Software Personnel (FSP) impacts of the entered CORADMO drivers projected over time. Impacts on all three variables are shown for each stage: Inception (I), Elaboration (E), and Construction (C). The CORADMO drivers impact both effort and schedule, often to the same extent. The third variable's (FSP) values are then simply the result of dividing effort (in person months) of a stage by its duration (in months).

Following the RAD impact per stage charts are charts showing of the totals of schedule and effort across stages. These represent the final results of the Technology Impact Analyzer. There are also charts comparing the results of both COCOMO-II.1998 and overall results. The data for the summary charts showing totals of schedule and effort across stages is on at the end of the page of "RAD Data".

O.4.4.1 CORADMO Effort and Schedule Impacts per stage

The first three pages of this work sheet display the effort, schedule or personnel impacts for each stage that result from the drivers values' change over time and technology. The impacts are shown in both tabular and chart form, with the chart always reflecting the "current" values of the drivers. An example is shown in Figure 10.

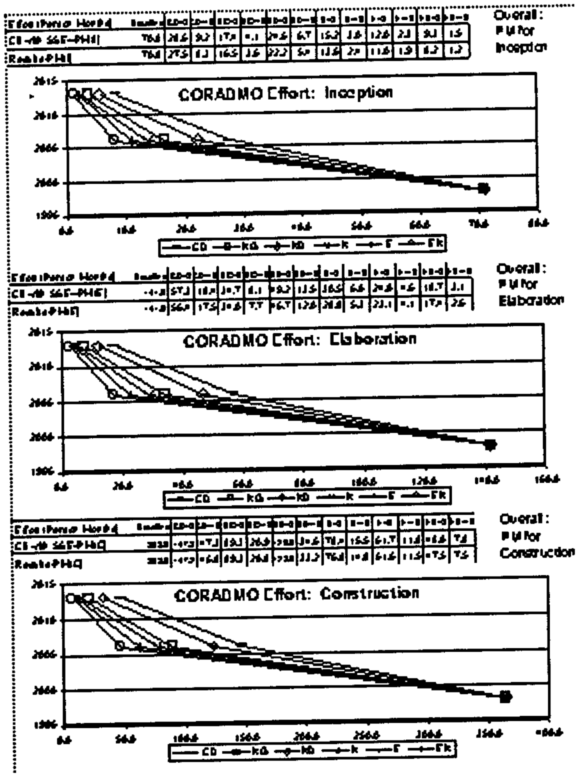


Figure 10. Impacts on Inception

The table above each of the charts shows the calculated results based on the COCOMOII calculations with the Staged Schedule and Effort (SSE) percentages applied. The "Results" row's values will contain the results based on the "current" driver values, and thus may have changes anytime there is input in the "new" row of the drivers. A single stage example is shown, as in Figure 11.

Effort (Person Months)	Baseline	CD+8	CD+15	KG+8	KG+15	KD+8	KD+15	K+8	K+15	E+8	E+15	EK+8	EK+15
CII with SSE--PM(I)	70.8	28.6	9.2	17.4	4.1	24.6	6.7	15.2	3.0	12.0	2.3	9.3	1.5
Results-PM(I)	70.8	27.5	8.3	16.5	3.6	22.2	5.4	13.6	2.4	11.0	1.9	8.2	1.2

Overall:
PM for
Inception

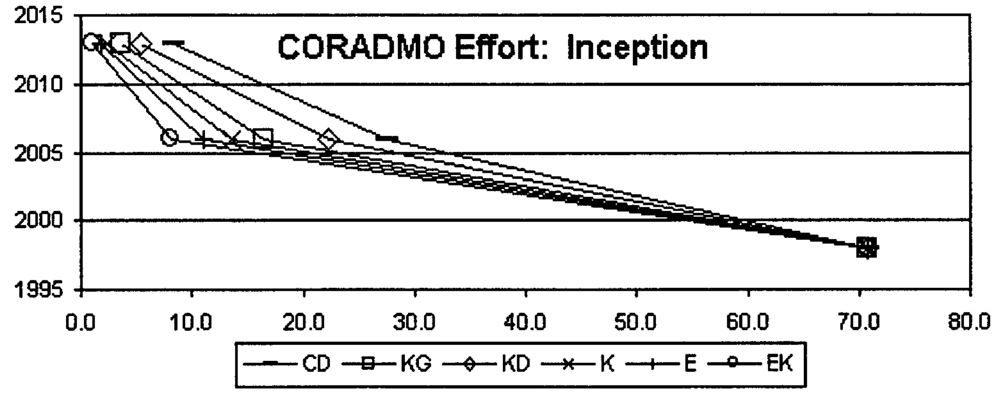


Figure 11. Combined COCOMO & CORADMO Impact on Effort for Inception

The remaining pages in the worksheet contain the summary results of the entire KBSA Technology Impact Evaluator. They are described in the next section.

O.5 Final Results: Technology Impacts Estimates

At the end of the "RAD Impact" worksheet, following the nine RAD impacts by stage charts, are the summary charts for effort and schedule by technology over time that result from the COCOMO-II.1998 and CORADMO driver changes over time. The "new/current" data for the summary charts is actually shown at the end of the "RAD Data" sheet.

There are three different types of charts:

1. Overall (effort or schedule for all three stages or just for development (elaboration plus construction), with some of these having alternative axes layouts;

2. COCOMO-II.1998 compared to CORADMO (final) results, with some of these charts showing only the major technology groupings (CD, K and EK);

3. Final results of default driver settings compared to new/current driver settings' results.

The list of all the charts corresponding to final results is shown below

Number	Title
1.	CORADMO Total Effort (effort on x axis)
2.	CORADMO Total Effort (years on x axis)
3.	CORADMO Total Effort (only for CD, K and EK)
4.	CORADMO Development (E+C) Effort with CoCoMo II Development (E+C) Effort
5.	CORADMO Development (E+C) Effort with CoCoMo II Development (E+C) Effort (only for CD, K and EK)
6.	CORADMO Total Schedule (schedule on x axis)
7.	CoRADMo Development (E+C) Schedule with CoCoMo II Development (E+C) Schedule(only for CD, K and EK)
8.	CoRADMo Development (E+C) Schedule with CoCoMo II Development (E+C) Schedule
9.	New/Current CORADMO Total (I+E+C) Effort with Default CORADMO Total (I+E+C) Effort
10.	New/Current CoRADMo Total (I+E+C) Schedule with Default CoRADMo Total (I+E+C) Schedule

O.5.1 Total Effort

The effort and schedule results are generated by adding the effort or schedule, respectively, for all three stages. Figure 12 shows the total effort after applying the Staged Schedule and Effort distribution percentages, and the COCOMO-II.1998 and CORADMO drivers

Total (I+E+C) Effort	Baseline	CD+8	CD+15	KG+8	KG+15	KD+8	KD+15	K+8	K+15	E+8	E+15	EK+8	EK+15
PM	576.3	231.1	72.6	139.8	32.1	192.8	50.6	119.2	22.5	95.8	17.5	73.0	11.3

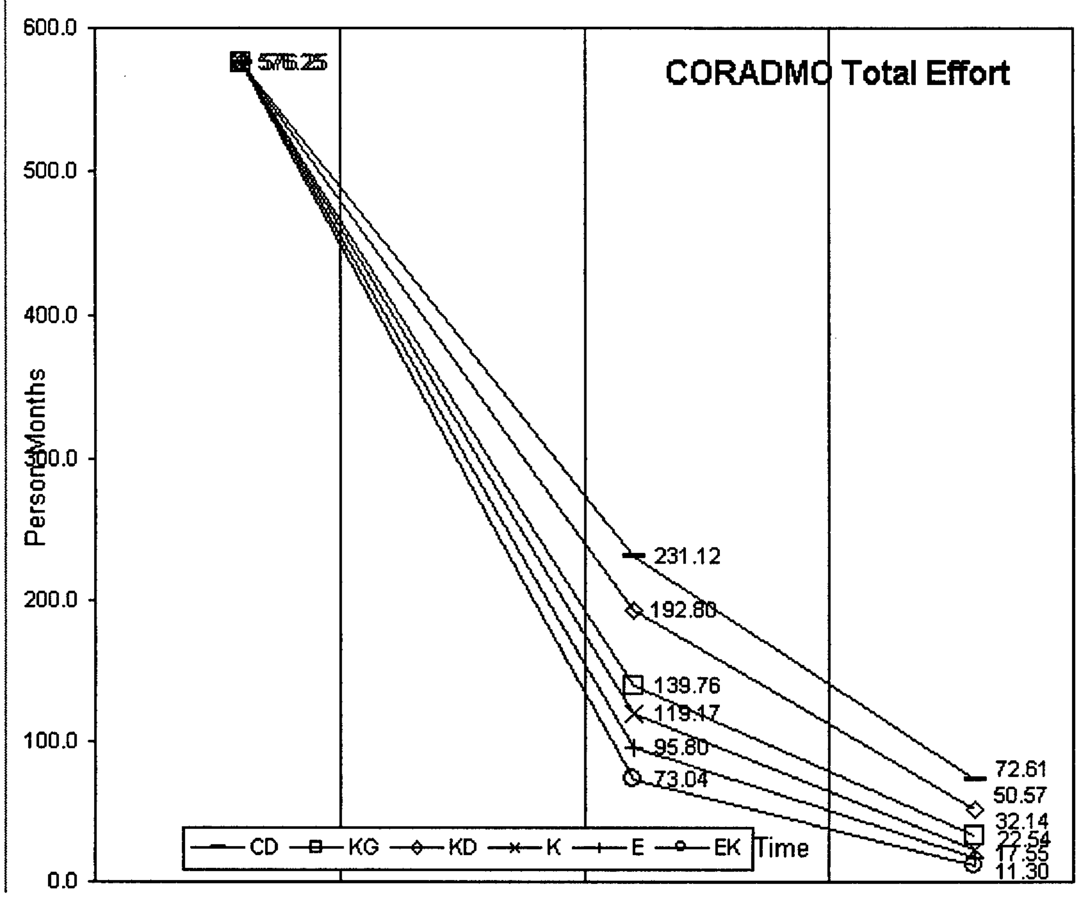
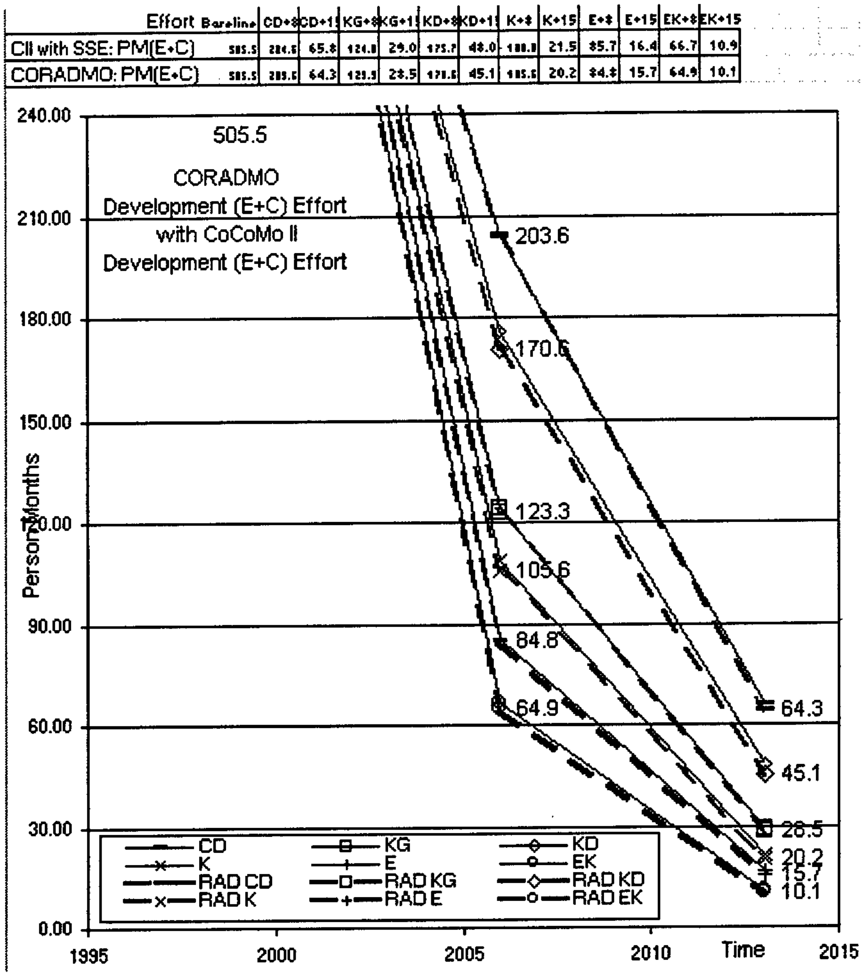


Figure 12. Total Effort after applying both COCOMO-II.1998 & CORADMO Drivers

In this part of the worksheet, the table above the charts shows the calculated results based on the updated values. The "updated values" are those based on the "current" ("default" otherwise "new" if modified) values of both the COCOMO-II.1998 & CORADMO drivers projected over time and technology-type.

O.5.2 COCOMO-II.1998 comparison with final, CORADMO results

Since COCOMO-II.1998 only calculates the effort and schedule for development, a second set of summary charts was generated so the COCOMO-II model results could be easily compared to the CORADMO model results. The second set of charts totals effort and schedule only for the Elaboration and Construction stages. Along with each chart are copies of the rows of the appropriate data from "CoRADMo Data" sheet. Figure 13 shows one of the comparisons of COCOMO-II.1998 only results and the final CORADMO results.



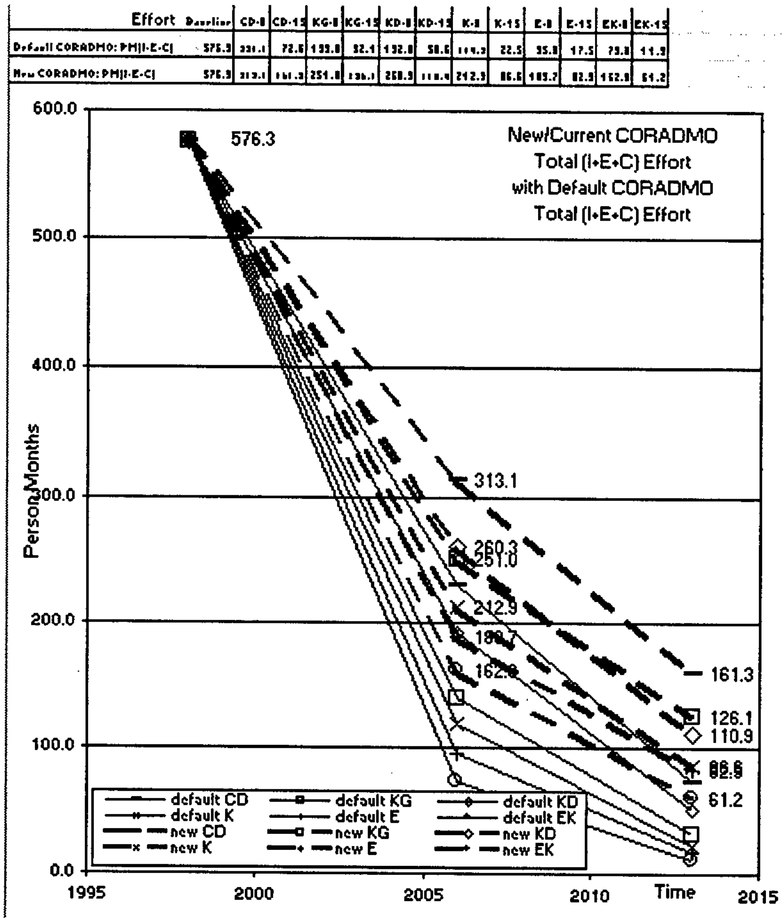
Effort	Baseline	CD+8	CD+15	KG+8	KG+15	KD+8	KD+15	K+8	K+15	E+8	E+15	EK+8	EK+15
CII with SSE: PM(E+C)	505.5	204.6	65.8	124.8	29.0	175.7	48.0	108.8	21.5	85.7	16.4	66.7	10.9
CORADMO: PM(E+C)	505.5	203.6	64.3	123.3	28.5	170.6	45.1	105.6	20.2	84.8	15.7	64.9	10.1

Figure 13. One of the comparisons of COCOMO-II.1998 only results and Final Results

Here, both the COCOMO-II.1998 set of calculations and the final results calculations are shown in the table above the chart. Again, the final results row's values will contain the results based on the "current" CORADMO driver values, and thus may have changes anytime there is input in the "new" row of the drivers. While only the data associated with the top row of the table, which contains the COCOMO-II.1998 calculation results, is shown in the chart, the final results' values are evident due to the dashed lines appearing in the chart.

O.5.3 New/current comparisons with default driver settings

Finally, a third set of charts is provide to provide a comparison of the overall effort and schedule results using the default driver values and the new/current driver values. This set of two charts is intended to assist with the use of the tool in sensitivity analysis studies. Along with each chart are copies of the rows of the appropriate data from "CoRADMo Data" sheet. Figure 14 shows a comparison of final CORADMO results for default and new drivers (with the only driver change being SIZE (change amount reduced by 50%). If there has been no change in any of the drivers, the lines will be coincident and only six will show on the chart.



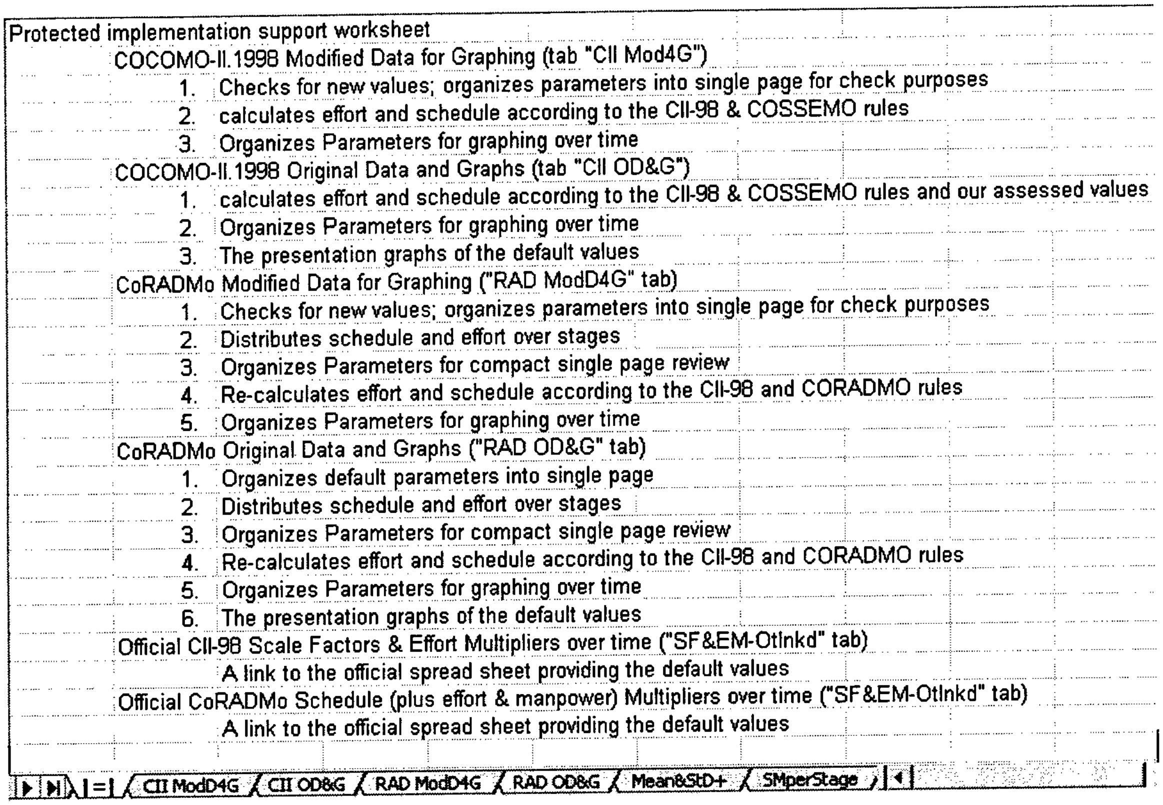
Effort	Baseline	CD-8	CD-15	KG-8	KG-15	KD-8	KD-15	K-8	K-15	E-8	E-15	EK-8	EK-15
Default CORADMO: PM(I-E-C)	576.3	231.1	72.6	133.8	32.1	132.8	58.6	114.3	22.5	35.8	17.5	73.8	11.3
New CORADMO: PM(I-E-C)	576.3	313.1	161.3	251.0	126.1	260.3	110.9	212.9	86.6	189.7	82.9	162.8	61.2

Figure 14. Comparisons of Effort Final Results for Default and New Drivers

Here, both the default and new final results calculations are shown in the table above the chart. Again, the new final results row's values will contain the results based on the "current" CORADMO driver values, and thus may have changes anytime there is input in the "new" row of the drivers. While only the data associated with the bottom row of the table, which contains the new calculation results, is shown in the chart, the default results values are evident due to the solid lines appearing in the chart.

O.6 Implementation

The workbook has six protected sheets which are used for the detailed layout of the drivers to facilitate the graphing shown in the 'Drivers' sections. These sheets also include sheets for the default values (i.e. the USC Center for Software Engineering assessed values) of the COCOMO-II.1998 and CORADMO drivers. Figure 15 provides details on the protected implementation sheets.



Protected implementation support worksheet

COCOMO-II.1998 Modified Data for Graphing (tab "CII Mod4G")

1. Checks for new values; organizes parameters into single page for check purposes
2. calculates effort and schedule according to the CII-98 & COSSEMO rules
3. Organizes Parameters for graphing over time

COCOMO-II.1998 Original Data and Graphs (tab "CII OD&G")

1. calculates effort and schedule according to the CII-98 & COSSEMO rules and our assessed values
2. Organizes Parameters for graphing over time
3. The presentation graphs of the default values

CoRADMo Modified Data for Graphing ("RAD ModD4G" tab)

1. Checks for new values; organizes parameters into single page for check purposes
2. Distributes schedule and effort over stages
3. Organizes Parameters for compact single page review
4. Re-calculates effort and schedule according to the CII-98 and CORADMO rules
5. Organizes Parameters for graphing over time

CoRADMo Original Data and Graphs ("RAD OD&G" tab)

1. Organizes default parameters into single page
2. Distributes schedule and effort over stages
3. Organizes Parameters for compact single page review
4. Re-calculates effort and schedule according to the CII-98 and CORADMO rules
5. Organizes Parameters for graphing over time
6. The presentation graphs of the default values

Official CII-98 Scale Factors & Effort Multipliers over time ("SF&EM-Otlnkd" tab)

A link to the official spread sheet providing the default values

Official CoRADMo Schedule (plus effort & manpower) Multipliers over time ("SF&EM-Otlnkd" tab)

A link to the official spread sheet providing the default values

Figure 15. Details on the protected implementation sheets

®U.S. GOVERNMENT PRINTING OFFICE: 1999-510-079-81200

MISSION
OF
AFRL/INFORMATION DIRECTORATE (IF)

The advancement and application of information systems science and technology for aerospace command and control and its transition to air, space, and ground systems to meet customer needs in the areas of Global Awareness, Dynamic Planning and Execution, and Global Information Exchange is the focus of this AFRL organization. The directorate's areas of investigation include a broad spectrum of information and fusion, communication, collaborative environment and modeling and simulation, defensive information warfare, and intelligent information systems technologies.